Downloaded from orbit.dtu.dk on: May 05, 2023

DTU Library

CHEMSIMUL: A chemical kinetics software package

Kirkegaard, Peter; Bjergbakke, Erling; Olsen, Jens V.

Publication date:
2008

Document Version
Publisher's PDF, also known as Version of record

Link back to DTU Orbit

Citation (APA):
Kirkegaard, P., Bjergbakke, E., & Olsen, J. V. (2008). *CHEMSIMUL: A chemical kinetics software package*. Danmarks Tekniske Universitet, Risø Nationallaboratoriet for Bæredygtig Energi. Denmark. Forskningscenter Risoe. Risoe-R No. 1630(EN)

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Risø DTU
National Laboratory
for Sustainable Energy

Risø-R-Report

CHEMSIMUL:
A chemical kinetics software package

Peter Kirkegaard, Erling Bjergbakke, Jens V. Olsen

Risø-R-1630(EN)

Risø National Laboratory for Sustainable Energy
Technical University of Denmark
Roskilde, Denmark
December 2008

Author: Peter Kirkegaard, Erling Bjergbakke, Jens V. Olsen

Title: CHEMSIMUL: A chemical kinetics software package

Division: IT-Service

Risø-R-1630(EN)
December 2008

Abstract

CHEMSIMUL is a computer program system for simulation of chemical kinetics.

It can be used for modeling complex reactions in many contexts. CHEMSIMUL contains a translator module and a module for solving the resulting coupled nonlinear ordinary differential equations, and it has a module for verifying the mass balance.

Pulse radiolysis and irradiation chemistry can be studied, as well as chemistry of high level waste radiation where decaying isotopes and their affiliations are considered. Transport phenomena and heterogeneous chemistry can be simulated by using “exchange equations”. Such equations admit simulation of general dynamic systems, with or without chemistry. Other special features are Arrhenius calculation of reaction rates and tools for dealing with ionic strength.

Standard chemistry notation is used in input files, and any number of reactions and reactants are accepted. Flexible graphical features with plot expressions are available. Besides quantities entering the differential equations it is also possible to work with “refreshable parameters”, which are defined by algebraic expressions. A discussion of the mathematical implementation is given.

The main computer platform is theWindows PC. CHEMSIMUL is coded with a modern GUI (graphical user interface). However, a classical command-driven version is available, too. The two versions use a common format of the basic input file and of the output report.

Homepage: http://chemsimul.dk

ISSN 0106-2840
ISBN 978-87-550-3653-6 (Internet)

Contract no.:

Group's own reg. no.:

Sponsorship:

Cover :

Pages: 80
Tables: 6
References: 23

Information Service Department
Risø National Laboratory for
Sustainable Energy
Technical University of Denmark
P.O.Box 49
DK-4000 Roskilde
Denmark
Telephone +45 46774004
bibl@risoe.dk
Fax +45 46774013
www.risoe.dtu.dk

Contents

1 Introduction

The aim of this booklet is to document a simulation tool, CHEMSIMUL, and describe how it is used properly. It is not intended as a chemical textbook.

1.1 Acknowledgements

The CHEMSIMUL program system was created many years ago by the pioneering work of Ole Lang Rasmussen at DTU Risø in Denmark. Since then the program has evolved and expanded alongside with its practical use by chemists in Denmark and many other countries. In 1998 a new era began when DTU Risø and CEA Saclay in France initiated a long-term collaboration with the aim of updating CHEMSIMUL with many new features. For much inspiration in the CHEMSIMUL development process we are indebted to the staff of CEA Saclay. In particular Dr. Pascal Bouniol has made very important contributions, especially to Chapters 8, 9, and 10.

1.2 CHEMSIMUL scope and overview

Chemical transformations that occur in the natural environment as well as in industrial processes are in general very complex. Even in well designed laboratory experiments it may be difficult to study elementary chemical reactions without interference from simultaneous side reactions. For this reason computer simulation is a powerful option for analysing complicated processes in atmospheric chemistry, for air pollution and combustion problems, and for developing new technologies.

The program system CHEMSIMUL was developed at DTU Risø as the result of a close co-operation between chemists and applied mathematicians. It is a computerized chemical simulator with the following main components:

- Module for inputting reaction equations in chemical notation
- Automatic translator from chemical equations to differential equations
- Mass balance check
- Pulse radiolysis from external source
- Internal radiolysis from decaying isotopes
- Transport phenomena and heterogeneous chemistry modelled by “exchange equations”
- Solution of the system of ordinary differential equations
- Interfacing “refreshable parameters” with the simulation
- Ionic strength calculations
- Write-up of the kinetics differential equations
- Output routines (graphical and tabular)

Additional special features will be described later.

CHEMSIMUL simulates an ideal process with uniform distribution of reactants and external effects such as irradiation or titration over the entire reaction volume. This uniformity is not always obtainable in experiments and seldom in nature. In non-ideal cases the accuracy

of the simulation can be improved by subdivision and making an individual simulation of each section of the reaction volume.

The program is constructed to simulate homogeneous kinetics in monophase combined with transport into and out of the reaction zone. The material balance is maintained entirely by the concentration of species; the reaction volume must therefore be kept constant as a unit volume. If transport is involved, e.g. liquid-gasphase equilibrium, the second volume must be the same unit volume as the reaction volume.

Only zero-, first-, and second-order reactions are allowed. Reactions of higher order must be emulated by suitable first-and second-order reactions or by using exchange equations, see below. For third-order processes in gas phase, see Section 11.8.

CHEMSIMUL works perfectly well as a purely chemical kinetics simulator. It has, however, many extra facilities.

First, the program can simulate radiolytic processes coming from an external irradiation source either as a single pulse or a pulse train. This feature is described in Section 2.3. It can also simulate the radiolysis from families of decaying nuclear isotopes which may be useful in nuclear waste studies; this is discussed in Chapter 9.

Furthermore CHEMSIMUL can perform ionic strength calculations and thus model the salt effect for charged reactants (Section 10.3).

As a very important extension, CHEMSIMUL can deal with quite general transport processes and heterogeneous phenomena that is expressible in terms of first-order differential equations. This is done by using the socalled *exchange equations*. More details about this feature are found in Chapter 8. In fact CHEMSIMUL's exchange equations may also be used to formulate and simulate general dynamics systems with no chemistry at all.

When preparing input data for CHEMSIMUL, the reactions are written in normal chemical notation, see e.g. the sample case in Section 3.2. The simulation results will include the concentrations of the chemical species in the reaction system as functions of time. They are presented in tabular and graphical form.

1.3 CHEMSIMUL historics

We conclude this chapter by giving a few notes on the history of CHEMSIMUL.

Chemists at Risø and elsewhere have used CHEMSIMUL and its predecessors for many years as a simulation tool supporting their experimental work. Rasmussen and Bjergbakke [1] give a historical account of the development of software for kinetics simulation at Risø over several decades, right from the beginning in 1966 when analogue methods were still prevailing. They point out the close connection with establishment of numerical techniques for fast and accurate integration of "stiff" non-linear differential equations. In the chemists' language stiffness means that the kinetic system has a wide range of relaxation times for perturbations. Stiff methods for Ordinary Differential Equations (ODE) were introduced at Risø shortly after 1971 when Gear [2] published his DIFSUB code. They replaced the classical fifth-order Runge-Kutta methods. Eventually, the ODE software in CHEMSIMUL was again replaced, first by a revised DIFSUB, then by EPISODE (Hindmarsh and Byrne [3]), and still later by the LSODA program (Petzold [4] and Hindmarsh [5]), which after revisions in 1997 and 2003 is the solver used today, now under the name DLSODA.

It was early realized that the construction of the ODEs from the reaction equations, which was originally done by hand, should be automated, and this led to the development of the translation module in CHEMSIMUL. It was a design criterion for the system that the chemist could express the reaction processes to be studied in familiar chemical nomencla-

ture.

The early versions of CHEMSIMUL were written in the computer language ALGOL. The old documentation by Rasmussen and Bjergbakke [1] relates to this language. Since 1986 PASCAL and MODULA-2 were in use as programming languages for mainframe versions, but from 1992 CHEMSIMUL has been entirely FORTRAN based. At that time it was implemented for PCs with a support granted from the IAEA. In 1997 it was converted to FORTRAN 90 (later FORTRAN 95) with the result that most of the restrictions on problem size were alleviated. The Windows PC is the predominant computer platform for running CHEMSIMUL at Risø and elsewhere.

Many users will appreciate that there is now a truly interactive CHEMSIMUL version for Windows PCs with a Graphical User Interface (GUI). This was made possible by a major coding effort in 2006–2008.

However, the CHEMSIMUL program is still available in command mode. This so-called "classical version" is retained mainly for reasons of documentation, program development, and versatility; it may run under any system that supports FORTRAN 95 [6], as for example Linux.

1.4 Organisation of booklet

The rest of this document is structured as follows: In Chapter 2 an outline is given of some concepts from physical chemistry that is important for CHEMSIMUL users. The transformation of chemical reactions to differential equations is described in Chapter 3, and an overview of the output facilities is presented in Chapter 4. Some additional CHEMSIMUL capabilities are mentioned in Chapter 5, while Chapters 6 and 7 give prescriptions for making proper input to the program. This is supplemented by Chapter 8 which describes the important feature of exchange equations. Chapter 9 contains discussions and input descriptions for radiolysis from decaying isotopes, which finds applications in the field of nuclear waste technology. Tools for making ionic strength calculations are found in Chapter 10, and Chapter 11 gives a number of practical hints for using the program. Reference to earlier simulations and papers, where CHEMSIMUL played an important role, can be found in Chapter 12. Documentation of the mathematical and numerical methods in CHEMSIMUL is provided for in Chapter 13, and in Chapter 14 we conclude with a list of the development and update historics for the program.

2 Some physico-chemical concepts

CHEMSIMUL provides a number of capabilities that are related to physical chemistry. In this chapter we shall briefly mention a number of physico-chemical concepts in the context of their usage in the program. Readers who have no interest in specific sections may just skip these. We begin with a survey of units and physical constants related to CHEMSIMUL.

2.1 Common units and physical constants

For the most part CHEMSIMUL allows a free choice of units in the computations, although the program originally was designed for kinetics in condensed phase with the traditional units which are: Concentration in $mol \times dm^{-3}$, rate constants in s^{-1} and $mol^{-1} \times dm^3 \times s^{-1}$ for first and second order reactions, respectively, activation energy in $kcal \times mol^{-1}$, heat of

reaction in $kcal \times mol^{-1}$, and specific heat capacity in $kcal \times mol^{-1} \times K^{-1}$. But CHEMSIMUL can be used with other units as well, for example the standard gas kinetics units with concentration in $molecules \times cm^{-3}$ and rate constant in $molecules^{-1} \times cm^3 \times s^{-1}$ for second order reactions. Time is always measured in s, and the temperature of the reaction system is always in degrees Kelvin (K). For convenience we state below some common units and physical constants that may be useful when working with CHEMSIMUL [7]:

Units:

Energy and heat:	$1\,J = 10^7\,erg$
	$1\,eV = 1.602176462 \cdot 10^{-19} J$
	$1\,kcal = 4184\,J$
Absorbed dose:	$1\,Gy\,(\text{gray}) = 1\,J \times kg^{-1} = 10^{-3} J \times g^{-1} = 10^{-1}\,krad$
	$1\,krad = 10^5 erg \times g^{-1} = 10^{-2} J \times g^{-1} = 2.390057 \cdot 10^{-3}\,kcal \times kg^{-1}$

Constants:

Avogadro's number:	$N_{\mathrm{A}} = 6.02214199 \cdot 10^{23}$ $(1\,mol = N_{\mathrm{A}}\,molecules)$
Gas constant:	$R = 8.314472\,J \times mol^{-1} \times K^{-1} = 1.987207 \cdot 10^{-3}\,kcal \times mol^{-1} \times K^{-1}$

We have collected some frequently used CHEMSIMUL units in Table 1 below, where each line represents a consistent set of units. The second line contains the corresponding input items, see Chapter 7.

Concentration	Rate Constants		Gas Constant	Activation Energy	Heat of Reaction	Specific Heat Capacity
CON	1.order	2.order	R	EA	Q	HCV
$mol \times dm^{-3}$	s^{-1}	$mol^{-1} \times dm^3 \times s^{-1}$	$J \times mol^{-1} \times K^{-1}$	$J \times mol^{-1}$	$J \times mol^{-1}$	$J \times mol^{-1} \times K^{-1}$
$mol \times dm^{-3}$	s^{-1}	$mol^{-1} \times dm^3 \times s^{-1}$	$kcal \times mol^{-1} \times K^{-1}$	$kcal \times mol^{-1}$	$kcal \times mol^{-1}$	$kcal \times mol^{-1} \times K^{-1}$
$molecules \times cm^{-3}$	s^{-1}	$molecules^{-1} \times cm^3 \times s^{-1}$	$J \times mol^{-1} \times K^{-1}$	$J \times mol^{-1}$	$J \times molecules^{-1}$	$J \times molecules^{-1} \times K^{-1}$
$molecules \times cm^{-3}$	s^{-1}	$molecules^{-1} \times cm^3 \times s^{-1}$	$kcal \times mol^{-1} \times K^{-1}$	$kcal \times mol^{-1}$	$kcal \times molecules^{-1}$	$kcal \times molecules^{-1} \times K^{-1}$

Table 1. Consistent sets of units for CHEMSIMUL

2.2 Yields of chemical species from radiation

An important feature of CHEMSIMUL is its ability to simulate radiolytic production of chemical species. Such radiolytic processes are zero-order reactions, i.e. they proceed independently of the concentrations of the chemical species. The radiation source may come from external pulse irratiation (Section 2.3), say from electron beams or γ-rays. Or it may come from internal isotopic radiation (Chapter 9).

For the standard situation of a dilute aqueous solution with density ρ_{liq} approximately equal to $10^3 kg/m^3$ CHEMSIMUL provides an automatic calculation of chemical species yields. Assuming that a species X, say $\mathrm{X} = \mathrm{H}_2$, is produced by the radiation, the rate of primary production is proportional to the dose rate:

$$\left(\frac{\mathrm{d[X]}}{\mathrm{d}t}\right)_{\mathrm{prim}} = c\,G_{\mathrm{X}}\,D'(t) \tag{1}$$

In this formula G_{X} is the *primary yield* (or "G-value"), which is always given in units of *molecules* per $he\,V$ where $1\ he\,V = 100\,eV$. Primary yields can be negative as well as positive.

For instance, when the radiation produces species with positive G-values, a corresponding negative value for $G_{\mathrm{H_2O}}$ should be included in order to preserve the mass balance. See also Section 11.4.

Moreover, $D'(t)$ in (1) is the absorbed dose rate in the solvent (water) measured in either Gy/s or $krad/s$, and c is a *conversion factor* giving the result in the unit $mol \times dm^{-3} \times s^{-1}$ for $(\mathrm{d[X]}/\mathrm{d}t)_{\mathrm{prim}}$. CHEMSIMUL has a unit selector, which allows the user to choose either Gy or $krad$ as the dose unit. Normally c is a constant which is computed automatically by the program from the selected dose unit. If this is Gy, then the value of c is $1.036427 \cdot 10^{-7}$, while $krad$ implies $c = 1.036427 \cdot 10^{-6}$. Thus we have:

$$\text{Dose unit } Gy \quad \Rightarrow \quad c = 1.036427 \cdot 10^{-7} \tag{2}$$

$$\text{Dose unit } krad \quad \Rightarrow \quad c = 1.036427 \cdot 10^{-6} \tag{3}$$

For more details on the evaluation of c the reader is referred to Section 2.10, where also the possibility of varying c during the simulation is mentioned.

2.3 External pulse and pulse-train irradiation

CHEMSIMUL is tailor-made to simulate a rectangular source pulse of finite time duration of the doserate:

$$D'(t) = \begin{cases} D_0 & 0 < t < t_{\mathrm{r}} \\ 0 & t > t_{\mathrm{r}} \end{cases} \tag{4}$$

where $t_{\mathrm{r}} =$ `RADTIME` is the time of irradiation.

The program also admits a train of such pulses, as shown in Figure 1.

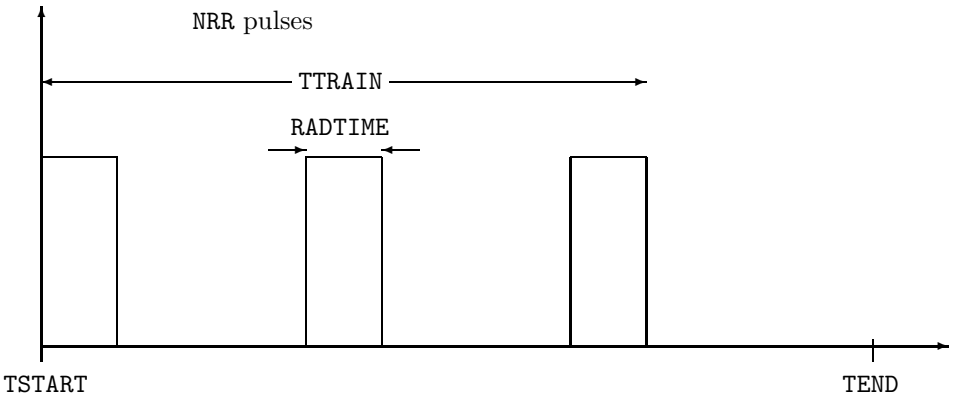

Figure 1. Train of irradiation pulses

The first pulse always begins at the same time as the simulation itself. This initial time is normally $t_{\mathrm{start}} = 0$ but may also be given an input value $t_{\mathrm{start}} =$ `TSTART`, cf. Section 7.6.

With reference to Section 7.8, the pulse train consists of $n_{\mathrm{rr}} =$ `NRR` individal pulses each of duration $t_{\mathrm{r}} =$ `RADTIME`. The duration of the total pulse train is $t_{\mathrm{train}} =$ `TTRAIN`.

2.4 Nuclear radiolysis

Radiolysis from decaying nuclear isotopes can also be studied by CHEMSIMUL. This important feature has been devoted an entire chapter in the booklet (Chapter 9).

2.5 Transport phenomena

As previously mentioned CHEMSIMUL can handle exchange equations for describing various forms of transport processes and other heterogeneous phenomena. Also here we reserve an entire chapter for this feature (Chapter 8).

2.6 Variation of reaction rate with temperature

The *Arrhenius equation* expresses the temperature dependence of the rate coefficient k in the exponential form

$$k = A \exp(-E_\mathrm{a}/(RT)) \tag{5}$$

where R is the gas constant, T is the thermodynamical temperature (K), and E_a is the *activation energy*. A is called the pre-exponential factor (or prefactor). It follows that

$$E_\mathrm{a} = RT^2 \frac{\mathrm{d} \ln k}{\mathrm{d}T} \tag{6}$$

CHEMSIMUL accepts a modified form of (5),

$$k = AT^\beta \exp(-E_\mathrm{a}/(RT)) \tag{7}$$

where the correction factor T^β is included for gas phase kinetics, β being an empirical exponent. This correction factor disappears when $\beta = 0$.

When (5) or (7) is applied, it is important to use compatible units of E_a and R, cf. Table 1 in Section 2.1. By default CHEMSIMUL sets $R = 8.314472\ J \times mol^{-1} \times K^{-1}$ which correponds to the unit $J \times mol^{-1}$ for E_a. Other units for E_a require other numerical values of the gas constant. This can be accomplished by the input command `R = ...`, cf. Chapter 7.

Apart from using the functional dependence $k = k(T)$ in (7) it is also possible to specify tables of k versus T and let CHEMSIMUL compute $k(T)$ by interpolation (Section 7.14.2).

2.7 Adiabatic processes

It is possible to simulate an adiabatic reaction at constant volume. This implies a variation of the temperature T which can be simulated by one extra equation. The source of the heat comes from the chemical reactions. Heat produced by external pulse irradiation or by isotope radiation is not taken into account. The specific heat capacity, $c_v(\mathrm{R}_s)$, must be known for each species R_s, and the heat of reaction, q_r, must be known for each reaction r. In this context q_r is defined as $-\Delta H$ (H = enthalpy), which means that q_r is positive for an exothermic reaction.

The differential equation for T is

$$\frac{\mathrm{d}T}{\mathrm{d}t} = \frac{Q'(t)}{\sum_{s=1}^{N} c_v(\mathrm{R}_s)\,[\mathrm{R}_s]} \tag{8}$$

where N is the number of species in the system, and $Q'(t)$ is the total heat production rate which can be computed as

$$Q'(t) = \sum_{r=1}^{M} k_r [\mathrm{R}_j][\mathrm{R}_k]\, q_r \tag{9}$$

Here M is the number of reactions, and R_j and R_k are the second-order reactants attached to the reaction r with rate constant k_r. (For a first-order reaction we should set $[\mathrm{R}_k] = 1$.)

The heats of reaction q_r are entered in the input records for the corresponding reactions r as `Q = ...`, and the specific heat capacities $c_v(\mathrm{R}_s)$ are given for each species as `HCV(...)`

= ...; see Sections 7.4 and 7.15. Of course the units of q_r and $c_v(\mathrm{R}_s)$ must be compatible, cf. Table 1 in Section 2.1.

CHEMSIMUL will only make a temperature calculation if at least one heat capacity `HCV(X) = ...` is given in the input file, where `X` is a species with non-vanishing initial concentration.

If some of the reaction rates $k_r = k_r(T)$ depend on T either through the modified Arrhenius expression (7) or through input tables, then there will be a coupling between (8) and the kinetic differential equations. If, on the other hand, all the k_r are constant, then (8) will give no feedback to the kinetic equations.

2.8 Ionic strength and salt effect

CHEMSIMUL provides tools for estimating the ionic strength I of an aqueous solution. The ionic strength is responsible for the salt effect by affecting the rate constant between charged reactants. Chapter 10 gives detailed instructions on how you can make the program compute I and the salt effect.

2.9 Molality, molar mass, and CUC

In the standard setup CHEMSIMUL works with volumetric concentrations of the solutes. The molar concentration, or *molarity*, of a solute X is

$$\text{Molarity}: \qquad C_{\mathrm{X}} = [\mathrm{X}] = \frac{n_{\mathrm{X}}}{V_{\text{solution}}} \tag{10}$$

where n_{X} is the molar amount of the solute and V_{solution} the volume of the solution. Molar concentration is usually expressed in $mol \times dm^{-3}$.

However, in some applications, e.g. computation of the salt effect, it is more appropriate to express concentration in *molality* A_{X}, which refers to the molar amount of the solute divided by the mass of the solvent:

$$\text{Molality}: \qquad A_{\mathrm{X}} = \frac{n_{\mathrm{X}}}{m_{\text{solvent}}} \tag{11}$$

The unit of molality is $mol \times kg^{-1}$. In contrast to molarity, the molality is an intrinsic entity in the sense that it does not vary with temperature or pressure.

Consider now a solution consisting of a solvent, typically water, and a number of solutes. Then we may write symbolically:

$$\text{solution} = \text{solvent} + \sum_i \text{solute}_i \tag{12}$$

where solute_i refers to some species X_i. The density of the solution is called ρ_{liq} (kg/m^3 of solution). At this place we shall introduce the *molar mass* M_{X} of a species X which is its mass in kg per mol of X.

For the given solution we define the *Conversion of Unit of Concentration*, abbreviated CUC, to be the mass of the solvent (kg) divided by the volume of the solution (dm^3). It is not difficult to show that, with the given units,

$$\mathrm{CUC} = 10^{-3} \times \rho_{\text{liq}} - \sum_i M_{\mathrm{X}_i} C_{\mathrm{X}_i} \tag{13}$$

As (13) shows, CUC can be calculated from the concentrations when we know the density ρ_{liq} of the solution and the molar masses M_{X_i}. Moreover we have

$$A_{\mathrm{X}_i} = \frac{C_{\mathrm{X}_i}}{\mathrm{CUC}} \tag{14}$$

2.10 Advanced use of conversion factor c

In this section we shall give a more detailed discussion of the conversion factor c which enters the primary production formula (1) in Section 2.2. We have seen that in dilute aquous solutions it was feasible to consider c as a constant parameter whose value was either (2) or (3), depending on the dose unit. When the latter is given, CHEMSIMUL automatically selects the appropriate value of c.

However, in certain non-standard situations (2) or (3) does not apply. One example is when the radiation yield rate is measured in $molecules \times cm^{-3} \times s^{-1}$ rather than in $mol \times dm^{-3} \times s^{-1}$. This would give another constant value of c.

In other cases it is desirable to work with a model where c varies during the simulation; CHEMSIMUL can deal with this situation by using *refreshable parameters*, see `CONVERT` in Section 7.15, and also Section 7.11. Examples are long-duration simulations where the pure-water assumption cannot be maintained, because water exposed to air is contaminated by new species and becomes acid. Or the water may be contaminated by species created by radiolysis. In such cases the density of the irradiated volume may vary with time, and changes in temperature may also affect the density.

We shall first compute c by using the pure-water approximation. As the time unit is the same for both members of (1), this expression can for our purpose equally well be written

$$c = \frac{[\mathrm{X}]}{G_{\mathrm{X}} D} \tag{15}$$

Now suppose that the liquid is irradiated by a dose of $1\,Gy = 1\,J \times kg^{-1}$, whereby the species X is formed. Recalling that G_{X} expresses the primary yield in units of *molecules* per heV, the production of X in *mol* per *kg* liquid is

$$n = \frac{G_{\mathrm{X}}}{c_{\mathrm{en}}} \times \frac{1}{N_{\mathrm{A}}} \tag{16}$$

where

$$c_{\mathrm{en}} = \frac{1 heV}{1J} = 1.602176462 \cdot 10^{-17} \tag{17}$$

is an energy conversion factor and N_{A} = Avogadro's number, cf. Section 2.1. Before we can insert (16) in the numerator of (15) we should convert it from *mol* per *kg* to *mol* per dm^3:

$$c = \frac{n \times \rho_{\mathrm{liq}} \times (1 dm^3/1 m^3)}{G_{\mathrm{X}} \times 1} = \frac{10^{-3}\,\rho_{\mathrm{liq}}}{c_{\mathrm{en}} N_{\mathrm{A}}} \tag{18}$$

where $\rho_{\,\mathrm{liq}}$ is the density (kg/m^3) of the liquid. Assuming almost pure water at 4°C we have

$$\rho_{\mathrm{liq}} \approx \rho_{\mathrm{w}} \approx 1000\,kg/m^3 \tag{19}$$

giving

$$c = \frac{1}{c_{\mathrm{en}} N_{\mathrm{A}}} \tag{20}$$

which has the numerical value (1). (If the unit for dose were instead *krad* we just multiply (16) by 10 and obtain (2).)

For non-dilute solutions it may be necessary to include a correction factor of (20), due to the fact that absorbed dose refers to the energy deposition per unit mass of the *solvent* (water). It can be shown that this correction factor is CUC, the Conversion of Unit of Concentration introduced in (13) in Section 2.9. Thus we obtain

$$c = \frac{1}{c_{\mathrm{en}} N_{\mathrm{A}}} \times \mathrm{CUC} \tag{21}$$

By considering the CUC expression (13), in which ρ_{liq} is the density of the solution, we see that the two terms vary as a function of the amount of solutes, which affects ρ_{liq}, and of the temperature, which affects ρ_{liq} too as well as the solubility of solid phases.

When there is no solutes in the liquid phase, the sum in (13) disappears, and we obtain $\mathrm{CUC} = 10^{-3}\rho_{\mathrm{w}}$. If the temperature T varies, then $\rho_{\mathrm{w}} = \rho_{\mathrm{w}}(T)$ also varies slightly, and this implies a variable c even in the pure-water case.

There may be still other complications when evaluating the primary production of chemical species by radiolysis. For instance, the absorbed energy depends for a given exposed irradiation on the stopping power of the irradiated volume. You should also notice that the primary yields (G-values) are temperature dependent.

3 Producing differential equations

When explaining the translation in CHEMSIMUL from chemical reactions to differential equations we shall first use a very simple reaction scheme with only one reaction, and then a more realistic sample case from combustion.

3.1 Simple example with one reaction

Consider the reaction

$$\mathrm{R1} + \mathrm{R2} \rightarrow \mathrm{R3} + \mathrm{R4} \tag{22}$$

with 4 *species* (2 *reactants* and 2 *products*). We assume that the reaction proceeds according to the law of mass action with the rate constant k. Suppose also that the chemical medium is irradiated either by an electronic beam or by γ-rays, such that the species R2 and R3 are produced by this radiation with yields determined by $G(\mathrm{R2})$ and $G(\mathrm{R3})$, respectively, cf. Sections 2.2 and 2.3. Then the resulting differential equations for the concentrations are

$$\begin{aligned}
\frac{\mathrm{d}[\mathrm{R1}]}{\mathrm{d}t} &= -k[\mathrm{R1}][\mathrm{R2}] \\
\frac{\mathrm{d}[\mathrm{R2}]}{\mathrm{d}t} &= -k[\mathrm{R1}][\mathrm{R2}] + c\,G(\mathrm{R2})D'(t) \\
\frac{\mathrm{d}[\mathrm{R3}]}{\mathrm{d}t} &= k[\mathrm{R1}][\mathrm{R2}] + c\,G(\mathrm{R3})D'(t) \\
\frac{\mathrm{d}[\mathrm{R4}]}{\mathrm{d}t} &= k[\mathrm{R1}][\mathrm{R2}]
\end{aligned} \tag{23}$$

Here $[\cdot]$ denotes concentration, $D'(t)$ is the dose rate at time t (e.g. from decaying isotopes or from a pulse or pulse train), and c is the conversion factor (Section 2.2).

The reaction between R1 and R2 is a second-order reaction, while the radiolytic production of R2 and R3 are zero-order reactions. (CHEMSIMUL does not treat third-order reactions or higher directly, but these can sometimes be emulated by lower-order reactions, cf. Section 11.8.) We see that already a single chemical reaction equation is described by a system of Ordinary Differential Equations (ODE) that are non-linear in the concentrations. Starting from the time t_{start} (normally zero) with the initial values of the reactant concentrations $[\mathrm{R1}]_0$, $[\mathrm{R2}]_0$, $[\mathrm{R3}]_0$ and $[\mathrm{R4}]_0$ we can integrate the system up to some final time $t = t_{\mathrm{end}}$.

Very small chemical systems such as (22) can be simulated by making a direct write-up of the ODE system, but for larger systems this would be tedious and error-prone. Therefore CHEMSIMUL has a module for automatic translation of the chemical reactions to differential equations.

3.2 Sample case from gas phase combustion

Let us now consider a more realistic reaction system discussed in [8] for modeling a H_2–O_2 combustion process. This example will be used repeatedly as a sample case:

$$
\begin{array}{lrcl}
\text{(R1)} & \mathrm{H + H} & \rightarrow & \mathrm{H_2} \\
\text{(R2)} & \mathrm{H + O_2} & \rightarrow & \mathrm{HO_2} \\
\text{(R3)} & \mathrm{H + HO_2} & \rightarrow & \mathrm{OH + OH} \\
\text{(R4)} & \mathrm{HO_2 + HO_2} & \rightarrow & \mathrm{H_2O_2 + O_2} \\
\text{(R5)} & \mathrm{OH + OH} & \rightarrow & \mathrm{H_2O_2} \\
\text{(R6)} & \mathrm{H + OH} & \rightarrow & \mathrm{H_2O} \\
\text{(R7)} & \mathrm{OH + HO_2} & \rightarrow & \mathrm{H_2O + O_2} \\
\text{(R8)} & \mathrm{OH + H_2} & \rightarrow & \mathrm{H_2O + H}
\end{array}
\tag{24}
$$

The corresponding part of the CHEMSIMUL input file, with rate constants, reads:

```
RE1: H+H=H2; A=4.0E7
RE2: H+O2=HO2; A=4.5E8
RE3: H+HO2=OH+OH; A=6.5E10
RE4: HO2+HO2=H2O2+O2; A=2.0E9
RE5: OH+OH=H2O2; A=4.0E9
RE6: H+OH=H2O; A=1.0E10
RE7: OH+HO2=H2O+O2; A=6.0E10
RE8: OH+H2=H2O+H; A=4.0E3
```
(25)

We note the similarity with the chemical notation in (24). When processing a reaction system as this, CHEMSIMUL scans and "digests" all the kinetic equations, symbol by symbol. On encounter it tabulates every new species and includes its name in the current set of symbols. It also stores the reaction rates. After the scan phase an assembly phase is invoked. The technical details are discussed in Section 13.1. Here we shall only show the outcome of the translation process, where we for illustration use CHEMSIMUL's ODE printout feature (Section 4.3):

```
D(H)/DT   = - 2*K1*H*H - K2*H*O2 - K3*H*HO2 - K6*H*OH + K8*OH*H2
            + G(H)*DOSERATE
D(H2)/DT  =   K1*H*H - K8*OH*H2
D(O2)/DT  = - K2*H*O2 + K4*HO2*HO2 + K7*OH*HO2
D(HO2)/DT =   K2*H*O2 - K3*H*HO2 - 2*K4*HO2*HO2 - K7*OH*HO2
D(OH)/DT  =   K3*H*HO2 + K3*H*HO2 - 2*K5*OH*OH - K6*H*OH - K7*OH*HO2
            - K8*OH*H2
D(H2O2)/DT=   K4*HO2*HO2 + K5*OH*OH
D(H2O)/DT =   K6*H*OH + K7*OH*HO2 + K8*OH*H2
```
(26)

(We have assumed that besides the chemical processes there is a radiolytic production of `H`.) The complete CHEMSIMUL input data file for the sample case is shown in Section 7.1.

4 Output possibilities

When running CHEMSIMUL, the program will produce a result file. Moreover, it has facilities for presenting the output in graphical form. We shall give a short description of each of these features, illustrating with the sample data set in Section 7.1.

This applies to the GUI version as well as the classical command version of the program, although the GUI has some extra facilities, cf. the CHEMSIMUL home page `www.chemsimul.dk`. The sample results below were obtained with the classical code.

4.1 Result file

The name of the result file will be the name of the input file with the extension replaced by `.res`. The first part of the file is an echo of the CHEMSIMUL input data as prepared according to Chapter 7. A statistics summary is given which also includes the actual values of the conversion factor c and the gas constant R, when relevant. Then follows the integration results in tabular form. These comprise the concentrations of all reacting species and other relevant variables as functions of time.

By and large the CHEMSIMUL output is self-explanatory. We show below the result file for our H_2–O_2 combustion case in Section 3.2:

```
            ################################
            CHEMSIMUL COMPILED:    NOV 2008
            ################################

 INPUT DATA FILE: h2-o2.dat
 DATE OF COMPUTATION:  23-NOV-2008 19:48
 ---------------------------------------
 Simulation of an H2 - O2 explosion with the following parameters:
 5 mbar O2 + 100 mbar H2 + Ar to 1 atm. k8=4000.

 UNBALANCED REACTION SYSTEM NOT ALLOWED
 STATUS OF REACTION SYSTEM: BALANCED

 ========== INPUT DATA: CHEMICAL REACTION SYSTEM ===========

 RE1:H+H=H2;A=4.0E7
 RE2:H+O2=HO2;A=4.5E8
 RE3:H+HO2=OH+OH;A=6.5E10
 RE4:HO2+HO2=H2O2+O2;A=2.0E9
 RE5:OH+OH=H2O2;A=4.0E9
 RE6:H+OH=H2O;A=1.0E10
 RE7:OH+HO2=H2O+O2;A=6.0E10
 RE8:OH+H2=H2O+H;A=4.0E3

 ============= INPUT DATA: START CONCENTRATIONS ============

 CON(O2)=2.0E-4
 CON(H2)=4.0E-3

 =============== INPUT DATA: OUTPUT CONTROL ================

 NO. OF PRINTOUTS FOR T > 0       =         12
 NO. OF SIGNIFICANT OUTPUT DIGITS =          5
 NO. OF CHARACTERS IN OUTPUT LINE =         72
 IRRADIATION PRINTOUTS FOR T > 0  =          2
 INTEGRATION STATISTICS:                   OFF
 DIFFERENTIAL EQUATIONS PRINTOUT:          OFF
 NO SAVE FILE

 ============== INPUT DATA: PULSE RADIOLYSIS ===============

 NO. OF IRRADIATION PULSES        =          1
 TOTAL DOSE                       = 9.0000E+00
 IRRADIATION TIME FOR EACH PULSE  = 5.0000E-09
 TOTAL IRRADIATION PERIOD         = 5.0000E-09

 PRIMARY YIELDS
 --------------
 G(H)=1.0

 =========== INPUT DATA: INTEGRATION PARAMETERS ============
```

```
TIME AT BEGINNING OF SIMULATION  = 0.0000E+00
TIME AT END OF SIMULATION        = 1.0000E-03
RELATIVE INTEGRATION TOLERANCE   = 1.0000E-05
FIRST INTEGRATION STEP ATTEMPTED =  AUTOMATIC
MAXIMUM INTEGRATION STEP ALLOWED =   INFINITE
JACOBIAN TYPE INDICATOR          =          1

================== INPUT DATA: GRAPHICS ==================

PLOT EXPRESSIONS
----------------
PE1:HO2

PLOT COMMANDS
-------------
plot(pe1)


=================== PROBLEM STATISTICS ====================

NO. OF CHEMICAL REACTIONS        =          8
NO. OF START CONCENTRATIONS      =          2
NO. OF PLOT EXPRESSIONS          =          1
NO. OF PLOT COMMANDS             =          1
NO. OF HEAT CAPACITY VALUES      =          0
NO. OF MAXIMUM CONCENTRATIONS    =          0
NO. OF CHEMICAL SPECIES          =          7
NO. OF DIFFERENTIAL EQUATIONS    =          7
NO. OF KINETIC TERMS IN DIF.EQS. =         25
NO. OF PRIMARY YIELDS            =          1
NO. OF EXCHANGE EQUATIONS        =          0
NO. OF EXCHANGE SPECIES          =          0
NO. OF SYMBOLIC CONSTANTS        =          0
NO. OF REFRESHABLE PARAMETERS    =          0
NO. OF NUCLEAR REACTIONS         =          0
NO. OF ISOTOPES                  =          0

CONVERSION FACTOR C = 1.036427E-07


================ OUTPUT TABLES: CHEMISTRY =================

RESULT TABLE
------------
    TIME         H           H2          O2         HO2          OH
 0.0000E+00  0.0000E+00  4.0000E-03  2.0000E-04  0.0000E+00  0.0000E+00
 2.5000E-09  4.6634E-07  4.0000E-03  2.0000E-04  5.2463E-11  2.6824E-15
 5.0000E-09  9.3257E-07  4.0000E-03  2.0000E-04  2.0983E-10  3.4450E-14
 1.0000E-04  1.8016E-11  4.0000E-03  1.9953E-04  2.8021E-07  4.4012E-08
 2.0000E-04  1.7401E-12  4.0000E-03  1.9957E-04  2.2235E-07  9.8900E-09
 3.0000E-04  4.9531E-13  4.0000E-03  1.9959E-04  1.9783E-07  2.8072E-09
 4.0000E-04  1.5859E-13  4.0000E-03  1.9960E-04  1.8158E-07  8.9927E-10
 5.0000E-04  5.5287E-14  4.0000E-03  1.9961E-04  1.6875E-07  3.1425E-10
 6.0000E-04  2.0820E-14  4.0000E-03  1.9961E-04  1.5790E-07  1.1788E-10
 7.0000E-04  8.5033E-15  4.0000E-03  1.9962E-04  1.4845E-07  4.6990E-11
 8.0000E-04  3.4407E-15  4.0000E-03  1.9962E-04  1.4010E-07  1.9756E-11
 9.0000E-04  1.5449E-15  4.0000E-03  1.9962E-04  1.3266E-07  8.7091E-12
 1.0000E-03  8.7072E-16  4.0000E-03  1.9963E-04  1.2597E-07  4.0046E-12

RESULT TABLE
------------
    TIME        H2O2        H2O
 0.0000E+00  0.0000E+00  0.0000E+00
 2.5000E-09  4.4069E-21  1.0486E-20
 5.0000E-09  9.9567E-20  3.3042E-19
 1.0000E-04  3.2911E-08  2.7137E-07
 2.0000E-04  4.5271E-08  3.0505E-07
 3.0000E-04  5.4028E-08  3.1210E-07
 4.0000E-04  6.1201E-08  3.1401E-07
 5.0000E-04  6.7327E-08  3.1459E-07
 6.0000E-04  7.2654E-08  3.1479E-07
```

```
 7.0000E-04  7.7342E-08  3.1486E-07
 8.0000E-04  8.1502E-08  3.1489E-07
 9.0000E-04  8.5219E-08  3.1490E-07
 1.0000E-03  8.8561E-08  3.1490E-07


=============== END OF CHEMSIMUL COMPUTATION ==============

CPU TIME FOR CHEMSIMUL:         0.05 SEC.
```

4.2 Graphical output and plot expressions

The modern Windows-based GUI version of CHEMSIMUL has native built-in graphical procedures, while the classical program has an interface to the freeware graphics program *gnuplot* [9] which is in widespread use on many different computers.

CHEMSIMUL may produce a "Graphics Table File" with one column for time, and one column for each of the *plot expressions* that are specified in the input file.

Plot expressions (`PE`...) are further discussed in Section 7.13. A plot expression may be just a single species concentration, as `PE1:HO2` in our sample case. However, the concept is much more versatile, as it supports all the 5 operators $+$, $-$, $\times$, $/$, and `^` (exponentiation), together with the common mathematical functions and derivation. Plot expressions are a useful facility in situations when a researcher wants to plot a curve which is directly comparable with certain experimental results. An example is the measurement of extinction

$$E = (\epsilon_A[\mathrm{A}] + \epsilon_B[\mathrm{B}])\ell \tag{27}$$

where ϵ_A and ϵ_B are the extinction coefficients for species A and B, respectively, ℓ is the optical path length, and $[\cdot]$ denotes concentration.

The plot expressions themselves will only produce the Graphics Table File. To produce an actual plot, a `PLOT` command should be used (Section 7.13).

4.3 Formal printout of differential equations

The program can give a formal print-out of the differential equation system to be solved. This may be a useful tutorial facility for understanding reaction kinetics. An example of this was shown at the end of Section 3.2. The reaction rates are labelled `K`n corresponding to reaction `RE`n. Radiolytic zero-order reactions are recognized. The ODE print-out is activated by the command `DIFFEQ`, cf. Section 7.7.

5 Additional features

In the following a number of additional CHEMSIMUL features will be described.

5.1 The stoichiometric mass balance

The accurate solution of the differential equation system describing the chemical reactions requires an overall conservation of the chemical mass balance. The numerical integration scheme itself preserves the mass balance, cf. Section 13.2, so there is no need to check for mass balance continually. We use instead a static consistency check. If the check fails, the simulation is halted before integration; this practice has proved useful for detecting input

errors in the write-up of the reaction equations. CHEMSIMUL recognizes each species as an entity with a name. This means that a formal stoichiometric balance does not imply an atom-to-atom balance. But it is possible to perform a *partial* consistency check based on the *stoichiometric matrix* **A**. This matrix is defined formally in Section 13.1, but its meaning is clear from the following example, where the reactions of the H_2–O_2 combustion case (24) are written in matrix form as a balance equation:

$$\begin{pmatrix} -2 & 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & -1 & 2 & 0 & 0 \\ 0 & 0 & 1 & -2 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & -2 & 1 & 0 \\ -1 & 0 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 1 & -1 & -1 & 0 & 1 \\ 1 & -1 & 0 & 0 & -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} \mathrm{H} \\ \mathrm{H2} \\ \mathrm{O2} \\ \mathrm{HO2} \\ \mathrm{OH} \\ \mathrm{H2O2} \\ \mathrm{H2O} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{28}$$

It must be possible to satisfy this equation by a set of strictly positive values for the masses of all the elements of the species vector.

The mathematical implementation of the balance check is discussed in Section 13.2.

5.2 Check of electro neutrality

As will be explained in Section 6.3, CHEMSIMUL admits species names with charge designators as e.g. `FE[+++]` or `OH[-]`. Such a convention enables the program to check the electro neutrality in the individual reactions. If the electro neutrality is violated the simulation will be rejected. Like the mass balance check, this feature may be useful in catching errors in the input data file.

5.3 Maximum concentration values for species

From the computed simulation curves CHEMSIMUL may estimate maximum values for individual species, cf. the command `MAXCON` in Section 7.15. This is done by interpolation in the result tables (Section 13.6).

6 Using the program

This chapter and the following contain instructions about using the CHEMSIMUL program. Supplementary information on exchange equations, radiolysis from decaying isotopes, and refreshable parameters will be given in the subsequent chapters.

6.1 Running the program

You may find information about procurement, installation, and licensing of the interactive GUI version of CHEMSIMUL on the CHEMSIMUL home page:

`www.chemsimul.dk`

We expect that users of the "classical" command-driven code, who are already familiar with the present booklet, will not find it difficult to use the GUI version. On the other hand we urge new users to get acquainted with Chapters 6 and 7 even if you only plan

to use the GUI, since the two versions work with essentially the same input and output files. In this way we may consider the GUI version as an "advanced editor" for preparing data and carrying out simulations. For practical reasons we shall for the main part relate the following descriptions to the classical code, and refer GUI users to the supplementary website information on the CHEMSIMUL home page.

The main computer platform for running CHEMSIMUL is the Windows PC, although the classical version also runs under Linux. The program has been tested with several brands of the Windows operating system.

The name of the code file is `Chemsimul.exe` for the GUI and `chem.exe` for the classical version. No particular installation procedure is needed, apart from storing the executable file in a suitable folder in your computer. However, a CHEMSIMUL license file with a valid license key is required for the GUI version. The classical command-driven version is free and runs without license keys or files; it may also be freely redistributed. Consult the CHEMSIMUL home page.

On a Windows PC CHEMSIMUL is typically launched from a desktop icon. With the GUI code you may then use the screen menus for example to select and edit an input file, and afterwards carry out a simulation and view the graphical output.

The classical version begins by issuing the following request:

```
ENTER NAME OF INPUT FILE:
```

You then type the input file name, including its extension (normally `.dat`), e.g. `mycase.dat` (when the extension is `.dat` it may be omitted); the output file will then be named `mycase.res`. (With the classical code it is also possible to enter the input file as command line input, i.e. you may type `chem mycase`.)

The program checks all the input data, and if errors are found it gives an error message which should be sufficient for identifying the error.

CHEMSIMUL has a progress bar which estimates the current fraction of the simulation being already completed. This feature is useful for estimating the progress of the integration procedure for long simulations.

All CHEMSIMUL computations are performed internally in a precision amounting to about 16 decimal digits.

6.2 Graphics interface

When the CHEMSIMUL job is finished, the main results are written to the `.res` file described in Section 4.1. If there are plot expressions in the input file (cf. Sections 4.2 and 7.13), then the program will produce a *Graphics Table File* whose name will be the name of the input file with the extension replaced by `.tbl`.

The GUI version of CHEMSIMUL has its own native graphics procedures built into the program. These procedures are easily accessed from the screen menus.

On the other hand, the classical version has a link to the graphics program *gnuplot* [9], which is freeware and may be downloaded from the Internet. You should observe that apart from the Graphics Table File, CHEMSIMUL will also produce a command file `chemgnu.plt` which is ready for launching in *gnuplot*.

Alternatively, you may prefer to view the Graphics Table File by means of your own favorite graphics program.

6.3 Name rules for species and other entities

Chemical species are identified by their *names* in CHEMSIMUL. Referring to the sample case of Section 3.2, `H`, `H2`, `O2`, and `HO2` are examples of chemical species names. Another kind of species are the exchange species defined in Chapter 8. Other named entities are the symbolic constants mentioned in Section 7.9 and the refreshable parameters (RPs) in Section 7.11.

We shall now give the rules for forming the names of species, symbolic constants (without `#`), and RPs (without `<`). Apart from certain restrictions, these names can be chosen freely. The main rule is that they should be alphanumeric and begin with a letter; they may contain small and capital letters and are interpreted in a case-sensitive way. Internal blanks in names will be squeezed out during the processing. The maximum permitted length of a name is 24 characters. A name cannot be empty.

A few particular CHEMSIMUL names are *reserved*. They are listed in Table 2 and must not be used in other contexts.

Name	Meaning
`TIME`	Simulation time in s
`TEMP`	Temperature (K) of the system
`TEMP2`	Dual temperature (Section 7.14.3)

Table 2: Reserved names in CHEMSIMUL

Only the upper-case version of these names are reserved.

The totality of chemical and exchange species names, symbolic constants, and RPs, must all be distinct.

It is allowed to qualify the name by a pair of square brackets containing strings of either of the signs `+` or `-`, like `FE[+++]` or `OH[-]`. This feature enables CHEMSIMUL to check the electro neutrality (Section 5.2) and also to estimate the rate constant dependency on the ionic strength (Section 10.3). It is also permitted to use square brackets as numerical markers in names, like `A[0]` or `B[219]`; any string of the digits 0 – 9 is accepted. The two modes may be combined, but not within a single bracket pair. Thus `A[2][+]` would be legal but not `A[2+]`. Square brackets must be properly paired. At most one pair of brackets with either `+` or `-` can occur in a name, and at most one pair with digits. Empty brackets are illegal, and so is nested use of square brackets.

Normal parenthesis pairs `(...)` are also allowed, and they can even be nested in more than one level. The pairing must hold on all levels. Square brackets cannot occur within parentheses on any level, and vice versa. Some examples of names containing parentheses are given below:

- `Ca(OH)2[0]`: Neutral aqueous complex, to be distinguished from the solid phase `Ca(OH)2`.
- `Fe(OH)4[-]`: Mono-charged negative ion, allows a condensed and more traditional writing of the complex `FeOHOHOHOH[-]`.
- `Fe(OH)2[+]`: Mono-charged positive ion, immediately recognizable from the bi-charged positive ion `FeOH[++]`; here the writing of `FeOH2+` would be ambiguous.
- `Al4(Fe(CN)6)3(H2O)17`: This name has several pairs of parentheses, one of these being nested.

6.4 Regular expressions in CHEMSIMUL

CHEMSIMUL supports several kinds of *expressions*. Section 7.11 gives the definition of *refreshable parameters* (RPs), e.g.

```
<z1 = a + b*z2
```

In Section 7.13 the *plot expressions* are mentioned, with the following simple example:

```
PE2: TEMP-273.15
```

Finally, in Chapter 8, *exchange expressions* are discussed. An example is:

```
d(N2g)/dt  =  coef * ((H2g0+N2g0)^2-(H2g+N2g)^2)/(H2g+N2g)
```

Common for the RP definition after `=`, the plot expression after `:`, and the exchange equation after `=`, is that each of them is a so-called *regular expression*. The rules for constructing regular expressions are as follows:

Regular expressions are general arithmetic expressions, which may be composed of the named entities of Section 6.3. Blank spaces in expressions are insignificant. When a species name occurs, say `HO2`, it is understood that the actual concentration of the species is substituted. Chemical and exchange species may enter, and so may symbolic constants (without `#`) and RPs (without `<`). Moreover, the reserved names in Table 2 can be used in the same way as species.

Allowed operators are sum (`+`), difference (`-`), product (`*`), division (`/`), and exponentiation (`^`). Operations proceed from left to right except for repeated exponentiations which are evaluated from right to left. Parentheses may overrule the calculation order in the usual way. The multiplication operator cannot be omitted. Thus writing `(a+b)(c+d)` or `2a` is illegal; you must write `(a+b)*(c+d)` and `2*a`.

A number of standard mathematical functions are available, cf. Table 3:

Function name	Meaning
`cos`	cosine with argument in radians
`cosh`	hyperbolic cosine
`exp`	exponential function
`ln`	natural logaritm (base e)
`log10`	logarithm with base 10
`sin`	sine with argument in radians
`sinh`	hyperbolic sine
`sqrt`	square root
`tan`	tangent with argument in radians
`tanh`	hyperbolic tangent

Table 3: Mathematical functions in CHEMSIMUL regular expressions

Note that the exponentiation operator (`^`) requires positive arguments. If you just need square or cube operations, you may write `A*A` or `A*A*A`.

Using two consecutive operators as in `10^-(a+b)` is not allowed; you should instead write `10^(-(a+b))`.

Case rules: The reserved names in Table 2 must all be in upper case. The mathematical functions may be in either case.

Braced items in regular expressions can be used to include certain other entities that are constant during the simulation, e.g. you may write the following plot expression:

```
PE1: FE[+++] * (1 - TIME/{TEND})
```

This example contains the *command value* `TEND` (Section 7.6). The actual input value of `TEND` is simply substituted into the expression. The syntax requires that `TEND` be enclosed in curly braces {}. Two other commands, `TSTART` and `CONVERT` (Sections 7.6 and 7.15), can be used in the same way. Note that if an unassigned command is used in the expression, its default value applies. (This is 0 for `TSTART` and $1.036427 \cdot 10^{-7}$ for `CONVERT`). Moreover, a primary yield (G-value) can enter a regular expression, as for example in

```
PE3: {g(H2)} - 0.2*{TEND}
```

or

```
PE4: {CONVERT} * {GG(H2)} * {DOSERATEG}
```

(where an Isotope Block (Chapter 9) was used in the last example).

Braced items are case insensitive, apart from species names.

7 Input file

We shall now describe how a CHEMSIMUL input file is constructed. The description pertains to the GUI as well as the classical version of the program. Naturally, the input capabilities are richer and more versatile in the GUI version, cf. the CHEMSIMUL website `www.chemsimul.dk`. Moreover, the GUI has powerful editing tools for constructing and modifying the input file. Hence, in daily use you need not be concerned about the correct preparation of the input file. Nevertheless all CHEMSIMUL users should read the present chapter in order to get a general understanding of the capabilities and the functionality of the program.

The input file is just a standard text file containing a number of lines or records. No line can exceed 1023 characters in length. Only standard text characters (ASCII) are applied. Hence, if you are using the classical version, you may use any text editor to edit your CHEMSIMUL input files; in contrast the GUI version will as mentioned above take care of the file editing itself.

The input data contain *commands* that are identified by well-defined character strings. For example, the reaction equations always begin with `RE`, the start concentration of reactants always with `CON`, etc. CHEMSIMUL is case insensitive for command typing, but case sensitive for typing names of species and other entities. Blank spaces in commands are ignored.

You can insert comment lines freely in the input file; such lines begin with an asterisk (`*`). For example, you may use such comments to make commands inoperative. Blank lines are also considered as comments. Yet another way of stating comments is to write an exclamation mark (`!`) after any command, which means that only the text after `!` is considered as a comment. In particular `!` can be used instead of `*`. Examples of comments:

```
* This is a comment

*save

RE13:H+O[-]=OH[-];A=2E10 ! Bjergbakke and Draganic 1989
```

A CHEMSIMUL input file terminates with the command `$ENDDATA`; any records after `$ENDDATA` are ignored.

The input data file for CHEMISIMUL is organized in *data blocks*. A data block correponds to a screen tab in the GUI program. In subsequent sections we shall describe the contents of each block. To illustrate we begin by reproducing the complete input data file for the combustion sample test with pulse irradiation that produced the output in Section 4.1:

7.1 Sample data set

```
$Identification
Simulation of an H2 - O2 explosion with the following parameters:
5 mbar O2 + 100 mbar H2 + Ar to 1 atm. k8=4000.
$Chemical reactions
RE1: H+H=H2; A=4.0E7
RE2: H+O2=HO2; A=4.5E8
RE3: H+HO2=OH+OH; A=6.5E10
RE4: HO2+HO2=H2O2+O2; A=2.0E9
RE5: OH+OH=H2O2; A=4.0E9
RE6: H+OH=H2O; A=1.0E10
RE7: OH+HO2=H2O+O2; A=6.0E10
RE8: OH+H2=H2O+H; A=4.0E3
$Irradiation
TOTALDOSE=9.0 ! Gy
RADTIME=5.0E-9
G(H)=1.0
$Concentrations
CON(O2)=2.0E-4
CON(H2)=4.0E-3
$Output control
PRINTS=12
*DIFFEQ
RADPRS=2
$Integration
TEND=1.0E-3
$Graphics
PE1:HO2
plot(pe1)
$ENDDATA
```

7.2 General rules for data blocks

The sample case shown above has 7 data blocks. (The trailing `$ENDDATA` record is not considered as a data block.) Each block begins with a *block header* whose first character is `$`. In this case the block headers are `$Identification`, `$Chemical reactions`, and so on. Below we give general rules for making the data blocks:

There are altogether 13 data blocks available. They are initiated by block headers as follows:

```
$ Identification
$ Chemical reactions
$ Concentrations
$ Integration
```

```
$ Output control
$ Irradiation
$ Symbolic constants
$ Exchange equations
$ Refreshable parameters
$ Isotope
$ Graphics
$ Tabular data
$ Miscellaneous
```

As other CHEMSIMUL commands, these block headers are interpreted in a case-insensitive way, and blanks will be suppressed. Only the first 8 nonblank characters (including `$`) are significant. This means that you may write e.g. `$CHEMICAL KINETICS`, `$chemical system`, `$exchange`, `$refreshable quantities` etc., if you like.

Blocks may come in any order, and so may items within blocks (apart from a few exceptions to be mentioned later).

Except within tables, comment and blank lines are allowed anywhere in the data file, including prior to the first data block.

All blocks are optional; they may be omitted, or they may be empty.

7.3 The Identification Block

The entire content of the `$ IDENTIFICATION` block will be reproduced verbatim in the beginning of the output file. In particular comment lines are retained in the print. The first text line in the block is used for headlines in plots. Example:

```
$ Identification
Carbonation of pure water by air, 1 week.
Cylindrical beaker with gaseous sky.
EXPERIMENTAL CONDITIONS:
Gas volume = 0.001 m3.
Liquid volume = 0.001 m3.
Interfacial area liq/gas = 0.01 m2.
Height of water = 0.1 m.
```

7.4 The Chemical Reactions Block

The `$ CHEMICAL REACTIONS` block is the block which contains the system of chemical reaction equations.

Each reaction equation is prefixed with a unique identification `RE`n`:`, where `RE` means `R`eaction `E`quation and n is the equation identification number. The reaction equation terminates with a semicolon (`;`) and is followed by the thermodynamics constants `A`, `EA`, `B` (Section 2.6), and the heat of reaction q_r = `Q` (Section 2.7), separated by a comma (`,`). Here `EA`, `B`, `Q` are optional, but `A` is required; if `EA` and `B` are omitted, then `A` just means the reaction constant k. Interspersed blanks for enhancing readability are allowed. Example:

```
$ CHEMICAL REACTIONS
RE3: H[+]+OH[-]=H2O; A=2.35E13, EA=3
RE2: H+HO2=OH+OH; A=2.5E11, EA=1.9, Q=38.3
RE64: 2*OH=H2O2; A=6.0E9
```

Note that it is not necessary to number the reactions consecutively. Combined with the use of comment lines (∗), this flexibility may be exploited to mask out single reactions, or to establish data bases on certain reaction mechanisms.

The stoichiometric constants are integers, while the rate constants are real numbers; their values can be entered in "free format". CHEMSIMUL supports "scientific notation" for exponents, using the symbol `E` (or `e`) for the exponent, i.e. `1.4E11` means $1.4 \cdot 10^{11}$. Note that `E11` is an illegal specification, but `1E11` is okay. The maximum order of a chemical reaction to be simulated is 2, i.e. up to 2 reactants can be written on the left hand side of the reaction equations, but there are virtually no restrictions on the number of products on the right hand side. There are no bounds on the number of reactions or species.

The terms `A`, `EA` and `B` are related to the modified Arrhenius expression for the reaction rate given in Section 2.6, $k = AT^{\beta}\exp(-E_{\mathrm{a}}/(RT))$, where T is the temperature in K: `A` $= A$, `EA` $= E_{\mathrm{a}}$, and `B` $= \beta$. The factor T^{β} is included for gas phase kinetics, with β being an empirical exponent. A is the so-called frequency factor, and E_{a} the activation energy. When activation energies are used, you must ensure that the gas constant R is given in a unit which is compatible with E_{a}, cf. Section 7.15.

By using the phrase `A=TABLE` it is possible to use tables of rate constants versus temperature. This is explained in Section 7.14.2. It is also possible to set a reaction rate equal to a *refreshable parameter* (RP) by writing say `A = <z1`, see Section 7.11.

The term `Q` corresponds to the heat of reaction q_r for reaction r, as described in Section 2.7. Note that the heat of reaction and the heat capacity values (`HCV` in Section 7.15) must be in compatible units; see also Table 1 in Section 2.1.

Default values for `EA`, `B`, and `Q` are zero, while the input of `A` as previously mentioned is mandatory for all reactions. The four specifiers may come in any order.

7.5 The Concentrations Block

The `$ CONCENTRATIONS` block contains initial values of the concentration of species in the simulation system.

Each command has the form `CON(X) =` ... and gives the value of the start concentration for the species `X`. Example:

```
$Concentrations
con(O2) = 2.0e-4
con(H2) = 4.0e-3
```

Default values of all initial concentrations are zero. The concentration unit is $mol \times dm^{-3}$ or $molecules \times cm^{-3}$, cf. Table 1 in Section 2.1.

Both the initial concentrations of chemical species and exchange species (Section 7.10 and Chapter 8) should be given in this block.

7.6 The Integration Block

The `$ INTEGRATION` block contains commands related to the simulation and the numerical solution of the corresponding differential equations. Example:

```
$Integration
eps = 1.0e-6
```

```
fststp = 1.0e-8
hmax = 0.001
jt = 1
tend = 12
```

These commands have the following meaning (alphabetic order):

EPS

The command `EPS =` ϵ_{rel} gives the relative accuracy ϵ_{rel} in the integration routine (DLSODA). Its default value is $1.0 \cdot 10^{-5}$.

Hint: In most cases the computing time increases when `EPS` *is made smaller. Difficult cases may require adjustment of one or more of the input data* `EPS`*,* `FSTSTP`*,* `HMAX`*.*

FSTSTP

The command `FSTSTP =` h_0 gives the initial integration step h_0, measured in s. The solver has automatic step size control and is able to estimate its own initial step by default.

Hint: To see the actual first step, use the `MATHINFO` *command (Section 7.7). The experienced user may sometimes save time by setting a larger* `FSTSTP`*. On the other hand, difficult cases may require very small values of* `FSTSTP` *to prevent integration failures. In such cases it may also be necessary to adjust* `EPS` *and/or* `HMAX`*.*

HMAX

The command `HMAX =` $h_{\max}$ gives the maximum allowed length $h_{\max}$ of an integration step in s. By default $h_{\max} = \infty$ which means that the solver is free to use as large steps as it wants.

Hint: To see the maximal step actually taken by the code, use the `MATHINFO` *command (Section 7.7). The experienced user may sometimes want to set* `HMAX` *to enhance the precision. This may occasionally be needed to prevent integration failures. In such cases it may also be necessary to adjust* `EPS` *and/or* `FSTSTP`*.*

JT

Indicator for Jacobian type. As explained in Section 13.4, the integrator may need the Jacobian $\mathrm{d}\boldsymbol{f}/\mathrm{d}\boldsymbol{y}$ of the differential equation system. `JT = 1` is the standard choice and the default setting in CHEMSIMUL. Other possible choices are `JT = 2` and `JT = 3`. All three options are explained and discussed in Section 13.4.

TEND

The command `TEND =` t_{end} gives the end time t_{end} in s for the simulation. By default $t_{\text{end}} = 0$.

Note: `TEND` is an absolute time. It is not necessarily equal to the time duration of the simulation which is `TEND` − `TSTART`, cf. below.

TSTART

The command `TSTART =` t_{start} gives the initial time t_{start} in s for the simulation. `TSTART` must not be negative. By default $t_{\text{start}} = 0$.

Hint: `TSTART` *may be useful in applications involving restart.*

7.7 The Output Control Block

The `$ OUTPUT CONTROL` block contains commands which control the printed output. Example:

```
$Output control
derivative(OH,H2)
diffeq
dig = 3
linele = 132
mathinfo
printonly(H2O2,H,TEMP)
printonly_rp(rp1,rp2)
prints = 25
radprs = 5
save
```

These commands have the following meaning (alphabetic order):

DERIVATIVE

The command `DERIVATIVE(...)` enables the printing of selected derivatives in a special table after the normal result table. The selected species should be entered as a comma-separated list in parantheses. `TEMP`, when relevant, may be included in the list (upper case), and so may exchange species (Chapter 8). If no `DERIVATIVE` command is present, no derivative table is produced. Only one `DERIVATIVE` command is allowed.

Hint: Computing a numerical estimate of the derivative is more difficult than computing the quantity itself. To obtain a sufficient accuracy it may be necessary to choose a smaller value of `EPS` *than its default value* $1.0 \cdot 10^{-5}$*. Precision enhancement can also be accomplished by the* `HMAX` *command. See also Section 13.5.*

DIFFEQ

If this command is written in the input file, then the differential equation system will be printed. By default the system is not printed. See also Sections 4.3 and 3.2. Zero-order terms with G-values are included in the print-out (without the conversion factor c), but exchange equations (Chapter 8) are omitted.

DIG

This command defines the number of significant digits in the result table of CHEMSIMUL. By default `DIG = 5`. It is the format of the main output tables that is adjustable in this way. Other CHEMSIMUL results are presented in fixed formats.

LINELE

The command `LINELE =` ℓ defines the number of characters per line in the main result table (and the output generated by `DIFFEQ`). By default $\ell = 72$. This is also the least possible value.

MATHINFO

This command causes some statistics about the integration to be printed in the result file. Furthermore, an additional file `message.ode` is created by the solver with more details about the integration. By default this feature is disabled.

PRINTONLY

The command `PRINTONLY(...)` restricts the printing of species concentrations in the result table to certain selected species. The selected species should be entered as a comma-separated list in parantheses. `TEMP`, when relevant, may be included in the list (upper case). If no `PRINTONLY` command is present, then all species will be printed. Only one `PRINTONLY` command is allowed.

PRINTONLY_RP

The command `PRINTONLY_RP(...)` is analogous to `PRINTONLY(...)` but pertains to the printing of refreshable parameters (RPs) (Section 7.11).

PRINTS

The command `PRINTS =` n defines the total number of output lines in the result table after the initial time, such that $n+1$ lines are actually printed, with `PRINTS =` n print intervals. Each line contains the time and the simulation results belonging to that time. In case of irradiation pulse(s) `PRINTS` includes `RADPRS` (see below), i.e. $0 \leq$ `RADPRS` $\leq$ `PRINTS`. By default `PRINTS` $= 0$.

RADPRS

This command controls the number of lines printed during irradiation. For a single pulse `RADPRS=` n_{r} defines the number n_{r} of equal time intervals in the result table during the irradiation pulse. For a pulse train (`NRR` > 1) the total irradiation period `TTRAIN` is divided in `RADPRS=` n_{r} equal subintervals for printing (Section 2.3). Note: `RADPRS` must satisfy the condition $0 \leq$ `RADPRS` $\leq$ `PRINTS`. Default value is 0. See also `PRINTS`.

SAVE

This command causes the computed results to be written to a save file by the end of the simulation. You may use the save file as a tool for preparing a "restart by hand" by using copy and paste technique. (We have excluded the possibility of making automatic restarts in CHEMSIMUL, because it might be difficult for the user to keep track of the exact conditions for the restart simulation.) By default the save file will bear the same name as

the main input file, only with the extension replaced by `.sav`, but you can overrule this by giving another file name as a parameter to `SAVE`, for example:

```
save pp2.dmp
```

For thermodynamic simulations also the temperature `TEMP` will be saved. Some hints for exploiting the save facility to do manual restarts are given in Section 11.6.

The save file will have the following format:

```
*** CHEMSIMUL SAVE FILE FOR MANUAL RESTART
*** pp2.dat 30-MAY-2008 16:14
*** CASE:PAGSBERG TEST OF B AND Q;
***
*** KEY NUMBERS FOR SIMULATION:
*** TIME AT BEGINNING OF SIMULATION..= 0.000000E+00
*** TIME AT END OF SIMULATION........= 2.000000E-04
*** RELATIVE INTEGRATION TOLERANCE...= 1.000000E-05
*** CONVERSION FACTOR C..............= 1.036427E-06
***
*** FINAL CONCENTRATIONS = NEXT INITIAL CONCENTRATION BY COPY/PASTE:
CON(OH)  =  8.5774520818151815E-08
CON(P)   =  4.7532617219312779E-07
CON(AE)  =  0.0000000000000000E+00
CON(AR)  =  4.0001036426865207E-02
TEMP     =  5.1882847677544373E+02
***
*** ENDSAVE ... END OF SAVE FILE
```

Record 1 is a fixed header, record 2 contains the name of the mother input file and a time stamp, and record 3 is the same as the first record in the Identification Block (Section 7.3). Then some key numbers for the simulation follow. Note that the two time values correspond to `TSTART` and `TEND`, respectively, and are *absolute* times; their difference `TEND` − `TSTART` equals the duration of the simulation. Next all the final concentration records follow, possibly a `TEMP` record, and finally an `ENDSAVE` record.

Note that when you copy `CON` records from a save file to a new input file in order to do manual restart, you can either let these replace the old `CON` records, or you may insert the new records after the old ones (but still in the Concentrations Block). In the latter case the new `CON` values overrule the old values.

Also note that the phrase "concentrations" should here be understood in a general sence, since these may also refer to exchange species (Section 7.10 and Chapter 8).

If you are using an Isotope Block, a revised Isotope Block will be saved just before the `ENDSAVE` record, see Section 9.5.5.

7.8 The Irradiation Block

The `$ IRRADIATION` block contains various commands related to external pulse or pulse-train irradiation, cf. Section 2.3. Example:

```
$Irradiation
G(H) = 1.0
G(OH) = 2.0
```

```
nrr = 5
radtime = 1e-6
ttrain = 2.0e-5
totaldose = 3.7
```

These commands have the following meaning (alphabetic order):

G

This command is used for entering primary yields or 'G-values' for external pulse irradiation for one or more species. See the example above for the syntax.

The primary yields are used if the reaction system is irradiated e.g. by high-energetic electrons or by gamma rays. They were introduced in Section 2.2, and are in units of *molecules* per heV. See in particular formula (1). The primary yields default to 0 in CHEMSIMUL.

The G-value input has been generalized to refreshable parameters (RPs) (Section 7.11). Thus you may write

```
G(OH) = <goh
```

where `goh` is supposed to be an RP defined in the Refreshable Parameters Block, for example by the equation

```
<goh = 2.0 + 0.0015*TEMP
```

This will cause `G(OH)` itself to be refreshed during simulation. (You can not, however, use the direct form `G(OH) = 2.0 + 0.0015*TEMP`.)

When using this facility you should ensure that the mass balance is respected for the G-values, see Section 11.4.

Note that if you associate a primary yield with an RP, as in `G(H2) = <ghydrogen`, you cannot use the braced-item form `{G(H2)}` in a regular expression, as we did in the plot expression `PE3` in Section 6.4; this presupposed that `G(H2)` was assigned a numerical value. Instead you should write

```
PE3: <ghydrogen - 0.2*{TEND}
```

Chemical species as well as exchange species (Section 7.10 and Chapter 8) may be associated with G-values.

NRR

This is the number of individual pulses in a pulse train (Section 2.3). Default value is 1 in case of irradiation and otherwise 0. Thus the command is only needed when there are more than one pulse.

RADTIME

`RADTIME=` t_r gives the irradiation time t_r in s for the rectangular pulse (or for each pulse if there are more than 1, Section 2.3). Default value is zero.

TOTALDOSE

The command `TOTALDOSE =` D gives the total irradiation dose D in Gy during the irradiation time for the rectangular pulse (or the total train for more than one pulse, Section 2.3).

If you prefer to work with *krad* as the dose unit, you should use the command `DOSEUNIT` in Section 7.15.

TTRAIN

This command is used in case of pulse-train irradiation, see Figure 1 in Section 2.3. `TTRAIN` $= t_{\text{train}}$ is given in s and is equal to the time duration of the total pulse train, including the pulse pauses. See also the command `NRR`. `TTRAIN` is required when `NRR` > 1. When `NRR` $= 1$ `TTRAIN` is automatically set to `RADTIME` and need not be set by the user.

7.9 The Symbolic Constants Block

In order to improve integrity and readability of the input data and avoid repetition of numerical constants, CHEMSIMUL admits the use of *symbolic constants* which are names that are assigned fixed numerical values. The symbolic constants may enter all regular expressions, i.e. exchange equations (Chapter 8), plot expressions (Section 7.13), and definitions of refreshable parameters (Section 7.11).

Symbolic constants are declared in a `$ SYMBOLIC CONSTANTS` block. The name of each constant must comply with the name rules given in section 6.3 and must be preceded by the token `#` in its declaration. On the other hand, when the constant is used in expressions as mentioned above, the name is given without the `#`. Example:

```
$Symbolic constants
#coef1 = 1.05E-8
#coef2 = 2.13E-6
```

As explained in Section 10.3 CHEMSIMUL associates a particular meaning with certain symbolic constants, provided the `IONIC` command in Section 7.15 is set. These constants all begin with `MM` followed by species names; they represent *molar masses*, cf. Section 2.9. (Without `IONIC`, constants with these names can still be used, but without the special meaning.)

7.10 The Exchange Equations Block

The `$ EXCHANGE EQUATIONS` block contains input for the so-called *exchange equations*. These equations can be used for describing transport processes and other heterogeneous phenomena. They may also be used in the study of general dynamic systems. An example of such a block is given below:

```
$exchange equations
* Hydrogen convection
d(H2g)/dt  =  2.11E-8*H2g * ((H2g0+N2g0)^2-(H2g+N2g)^2)/(H2g+N2g)
d(H2ge)/dt = -2.11E-8*H2g * ((H2g0+N2g0)^2-(H2g+N2g)^2)/(H2g+N2g)
* Nitrogen convection
d(N2g)/dt  =  1.05E-8*N2g * ((H2g0+N2g0)^2-(H2g+N2g)^2)/(H2g+N2g)
d(N2ge)/dt = -1.05E-8*N2g * ((H2g0+N2g0)^2-(H2g+N2g)^2)/(H2g+N2g)
```

```
* Add neutral instructions for exchange species H2g0 and N2g0
d(H2g0)/dt = 0
d(N2g0)/dt = 0
```

On the left-hand side of each exchange equation the time derivative of some species concentration is stated in the syntax shown. This time derivative should be understood as the contribution to the production rate from some specific process. The expressions on the right-hand side are constructed as *regular expressions*. You should consult Section 6.4 where the syntax of regular expressions are given, and where all the possible constituent elements are mentioned. We call the species that occur only in the exchange equations *exchange species*, while the others are *chemical species*. Both kinds are allowed to occur on either side of an exchange equation. We stress again that an exchange equation is *not* itself a differential equation. It accounts for a particular contribution to a production rate for a species; other contributions may come from the chemical kinetics or other exchange equations. Exchange equations are allowed to be *additive*, which means that the same species `X` may occur on the left-hand side of more than one exchange equation. In such a case CHEMSIMUL adds the corresponding right-hand sides together and uses the sum as the total exchange contribution to `X`. This feature may also be useful in breaking up very long expressions into several right-hand side addents. For reasons of data check and integrity, CHEMSIMUL requires that each of the exchange species be mentioned on the left side of an exchange equation; if necessary "neutral instructions" must be given as shown in our example for `H2g0` and `N2g0`.

The concept of "concentrations" is formally retained for the exchange species too, even though their true units might be something else. With this interpretation, initial concentrations of exchange species can be defined exactly as for the chemical species by using `CON`, e.g.

```
CON(H2g0)=2.09E-8
```

Like for chemical species the initial concentrations of exchange species are 0 by default. Names of exchange species are formed in the same way as for chemical species (Section 6.3). Often the use of symbolic constants are useful in exchange equations. As an example we may take the first two equations shown above. Assuming we have defined the symbolic constant

```
#coefh = 2.11E-8
```

in the Symbolic Constants Block, we may write

```
d(H2g)/dt  =  coefh * H2g * ((H2g0+N2g0)^2-(H2g+N2g)^2)/(H2g+N2g)
d(H2ge)/dt = -coefh * H2g * ((H2g0+N2g0)^2-(H2g+N2g)^2)/(H2g+N2g)
```

More examples of the use of exchange equations are shown in Chapter 8.

7.11 The Refreshable Parameters Block

CHEMSIMUL supports the so-called *refreshable parameters* (RPs) which are defined by algebraic equations. Such equations are not themselves differential equations but are interlinked to the ODE (ordinary differential equation) system during the simulation. Their values are refreshed as the integration proceeds.

Refreshable parameters are useful in case of long-duration simulations when the model contains certain physical entities as for example the density ρ_{liq}, the molality, and the

CUC , which are supposed to change continually with time. See Sections 2.9 and 2.10 and Chapter 10.

In general it is not an easy task to solve combined sets of differential and algebraic equations (DAE); special methods and software are required for such systems (Asher and Petzold [10]). But if we impose certain restrictions on the RPs it is indeed possible to interface them to the ODE solver (in our case DLSODA). In CHEMSIMUL the refreshable parameters must be *a chain of explicit equations*. An *explicit* RP definition of say z_1 has the form

$$z_1 = f(z_2, z_3, \ldots; \text{other parameters}) \tag{29}$$

where z_2, z_3, ... are other RPs, and "other parameters" may include e.g. current species concentrations and symbolic constants. Implicit equations in the RPs are not supported by CHEMSIMUL.

The input write-up of the RPs should be made in the `$ REFRESHABLE PARAMETERS` block. We give a small example of a complete data file to show the syntax:

```
$identification
Artificial test case with refreshable parameters
$chemical reactions
RE1:R1+R2=R3;A=<z1
$concentrations
CON(R1)=4.1992D0
CON(R2)=1.6941D0
CON(R3)=0.8852D0
$integration
TEND=10
$output control
PRINTS=10
$symbolic constants
#a = 1.3816
#b = 0.2377
$refreshable parameters
<z1 = a + b*z2
<z2 = 1/R1 - a/b
$graphics
pe1: z1
pe2: z2
plot(pe1,pe2)
$enddata
```

We see that in the `$ REFRESHABLE PARAMETERS` block two RPs `z1` and `z2` are defined by preceding their names with the token `<`, and writing their defining expressions on the right-hand side. Each right-hand side may contain (chemical or exchange) species concentrations (here `R1`), temperature, symbolic constants (here `a` and `b`), and other RPs (here `z2`). Note that RPs on the right-hand side should *not* be preceded by `<`.

The example also illustrates that RPs may provide feedback to the chemical kinetics. In this case the formation of the species `R3` is enabled by equating the reaction rate to an RP `z1` using the following syntax:

```
RE1: R1 + R2 = R3; A = <z1
```

This syntax can not be extended to general expressions, i.e. writing `A = <z1 + <z2` would be illegal.

RP names should comply with the naming rules in Section 6.3. An RP expression must be a *regular expression*. You should consult Section 6.4 where the syntax of regular expressions are given, and where all the possible constituent elements are mentioned. In particular, an RP expression may contain other RPs.

Here is a valid example of a more complicated RP:

```
<G1 = 10^(-0.5092*(SQRT(IS)/(1+SQRT(IS))-0.3*IS))
```

(But note that the expression

```
<LOG10(G1) = -0.5092*(SQRT(IS)/(1+SQRT(IS))-0.3*IS)
```

would be invalid because it is not explicit, since `G1` occurs as the argument of a logarithm.)

The RP values obtained during simulation are printed in a separate output table. See, however, the command `PRINTONLY_RP` in Section 7.7.

As our test example shows, RPs are allowed to enter plot expressions. They may also occur in exchange equations.

RPs may also be utilized as a means of storing current accounts when computing expressions. This may give reductions in write-up as well as computing time. As an example, consider the two first exchange equations in the example of Section 7.9. These may be written

```
d(H2g)/dt  =  rhsH2g
d(H2ge)/dt = -rhsH2g
```

provided the RP definition

```
<rhsH2g = 2.11E-8*H2g * ((H2g0+N2g0)^2-(H2g+N2g)^2)/(H2g+N2g)
```

is included in the `$ REFRESHABLE PARAMETERS` block.

The RPs in the `$ REFRESHABLE PARAMETERS` block must form a *resolvable chain of explicit equations*, which means that it should be possible to evaluate all the RPs in turn by explicit formulas without ever encountering an RP that is not already defined. In particular, no RP definition must be "circular", i.e. it must not reference itself directly or indirectly. Finding a resolvable chain may require a reordering ("chaining") of the RP definitions. However, you need not be concerned with the order in which you mention the RPs in your input file; the program will find a proper permutation if necessary to do the ordering task. Details of CHEMSIMUL's mathematical handling of RPs are given in Sections 13.9 and 13.10.

7.12 The Isotope Block

The `$ ISOTOPE` block contains input for simulating radiolysis from decaying isotopes. Due to the importance of this feature, it will be described in detail in Chapter 9. Here we shall just list a real case example of such a block:

```
$ ISOTOPE

* Nuclear reactions, decay constants (s-1) and initial dose rates (Gy/s)
NR1:Pu241=Am241;    k=1.53066E-9,    DB=2.80E-11
NR2:Am241=Np237;    k=5.07626E-11,   DA=4.12E-5,    DG=1.70E-9
```

```
NR3:Cs137=Ba137m;   k=6.904574E-10,  DB=6.29E-5
NR4:Cs137=Ba137;    k=4.126939E-11,  DB=8.98E-6
NR5:Ba137m=Ba137;   k=4.526823E-3,   DG=2.15E-4

* Initial activities (Bq)
ACTIVITY(Pu241)=5.2E12
ACTIVITY(Am241)=5.0E10
ACTIVITY(Cs137)=1.755E13
ACTIVITY(Ba137m)=1.755E13

* Primary yields for alpha radiation (molecules/heV)
GA(H2)=1.30
GA(E[-])=0.06
GA(H)=0.21
GA(H[+])=0.06
GA(OH)=0.24
GA(H2O2)=0.985
GA(HO2)=0.22
GA(H2O)=-2.71

* Primary yields for beta radiation (molecules/heV)
GB(H2)=0.45
GB(E[-])=2.66
GB(H)=0.55
GB(OH[-])=0.10
GB(H[+])=2.76
GB(OH)=2.67
GB(H2O2)=0.72
GB(H2O)=-6.97

* Primary yields for gamma radiation (molecules/heV)
GG(H2)=0.45
GG(E[-])=2.66
GG(H)=0.55
GG(OH[-])=0.10
GG(H[+])=2.76
GG(OH)=2.67
GG(H2O2)=0.72
GG(H2O)=-6.97

* Specific isotopic dose rates
DOSERATEB(Pu241)
DOSERATEA(Am241)

* Decay print of all activities and amounts (not activated here)
* DECAYPRINT
```

7.13 The Graphics Block

The GUI version of CHEMSIMUL has richer graphical facilities than the classical program, see the website `www.chemsimul.dk`. However, the structure of the graphical input is the same in the two versions.

The `$ GRAPHICS` block may contain *plot expressions* and *plot commands*, cf. Section 4.2. Furthermore, there may be one or more *data file references* (`DF...`) for plotting external data files. Example:

```
$Graphics
PE1:HO2
PE2:OH
PE3:HO2 + OH
DF1:ho2resul.dat
plot(pe1,pe2,pe3)
plot(pe1,df1)
```

These commands have the following meaning (alphabetic order):

DF

Each data file reference is initiated by `DF` followed by a number and a colon. The numbers need not come in natural order but they must be distinct. Items like `DF1` are called *data file headers*. After the colon follows the name of a file containing a table, which typically contains experimental data.

The GUI version is much more flexible than the classical with respect to the interpretation of external data files. First, you should notice that the classical version requires the file to reside in the same folder as the executable program file and be referred by its local name; in constrast the GUI version has a full-fledged file reference system with complete path designations. Next, in the classical version only files with simple xy-tables in text format are allowed, while the GUI also supports multi-column table files in other formats e.g. Excel. You are advised to consult `www.chemsimul.dk`. Below we describe the restricted features accepted by the classical version.

The referenced file must contain pairs of data in two columns separated by an optional comma and at least one space. In the first column we must always have the time values (s), while the second contains the experimental (or other) data values. Example of such a file:

```
0.00000     0.00
0.00025     1.65e-7
0.00050     1.42e-7
0.00075     1.30e-7
0.00100     1.15e-7
0.00125     1.11e-7
0.00150     1.03e-7
0.00175     0.96e-7
0.00200     0.84e-7
```

To launch a plot with external data files, you should use the `PLOT` command.

PE

Plot expressions, which were mentioned in Section 4.2, will not by themselves create any plots, but only a Graphics Table File. To make the plots, you should use the `PLOT` command.

Each plot expression is initiated by `PE` followed by a number and a colon. The numbers need not come in natural order but they must be distinct. Items like `PE1` are called *plot-expression headers*. The expression after the colon must be a *regular expression*. You should

consult Section 6.4 where the syntax of regular expressions is given, and where all the possible constituent elements are mentioned. In this way plot expressions admit quite general arithmetic expressions in the species concentrations and other entities to be plotted as functions of time.

Plot expressions allow one extension of the syntax of regular expressions: You can handle derivatives of species (incl. `TEMP`), simply by appending a trailing `'` mark, as `H2O2'`. Thus the following plot expression is legal:

```
PE3: 210*H2O2'+1800*O2[-]-FE[+++]^1.5/220-TIME+log10(OH)
```

The remarks given as "Hint" for the `DERIVATIVE` command, apply here, too (Section 7.7).

PLOT

Plot commands are needed to produce actual plots based on the Graphics Table File. Each plot command creates one graphical *page* or *frame*. A single frame may contain several curves. The general structure of the plot command is:

`plot` (independent variable `;` dependent variables `:` options)

Thus we have three argument groups, 1, 2, and 3. The semicolon may be omitted which means that the indepedent variable defaults to `TIME`. Alternatively, you may explicitely state `TIME` (upper case), or one of the plot-expression headers, say `pe1`, as independent variable. The colon may also be omitted, meaning default options. Argument groups 2 and 3 may contain comma-separated lists of items. The items in group 2 are plot expression headers and data file headers in any order.

The possible options in the classical program are `XLOG`, `YLOG`, `MARK`, `GRID`. `XLOG` produces a logarithmic scale on the x-axis, and `YLOG` does the same on the y-axis; by default both axes are linear. `MARK` produces curve marks which might be useful for discerning curves in black-and-white representation, and `GRID` establishes a set of grid lines on the frame. Options are interpreted in a case-insensitive way. More options are available in the GUI version, see `www.chemsimul.dk`.

The capabilities of the `PLOT` command are illustrated by the following examples:

```
plot(pe1,pe2) ! plots pe1 and pe2 as functions of TIME; default options.
plot(TIME;pe1,pe2) ! same as above; note that TIME must be uppercase.
plot(pe1;pe2) ! classical xy-plot with pe1 as independent variable.
plot(pe5,pe2,pe3:grid,mark) ! plot with grid lines and curve marks.
plot(pe1,pe2,pe3:xlog) ! logarithmic TIME scale.
plot(pe1,pe4:ylog) ! logarithmic y scale i.e. for pe1 and pe4.
plot(pe1:xlog,ylog) ! logarithmic scales for both TIME and pe1.
plot(pe1,df1) ! co-plotting a plot expression with an experimental file.
```

When `TIME` is not the independent variable, the options `XLOG` and `YLOG` have no effect in the classical program version.

7.14 The Tabular Data Block

CHEMSIMUL accepts certain data in tabular form. We shall first give the general rules for writing input tables in CHEMSIMUL and then describe some particular kinds of tables.

7.14.1 General rules for tables

All tables must be placed in the `$ TABULAR DATA` block. We give below an example with two tables:

```
$tabular data
table temp re12
200   1.00E03
300   1.30E03
350   1.40E03
375   1.25E03
390   1.22E03
400   1.20E03
table  time temp
0       298.15
1       298.50
10      300.00
1E2     305.00
3.4E3 310.00
3.0E4 320.00
3.0E5 330.00
3.0E6 340.00
3.1E7 350.00
```

As the example shows, there may be several kinds of tables. Each table is initiated by a *header record*. The table itself is normally a standard xy table with x as the independent and y the dependent variable. There must be at least 2 records. Each record is a pair of values (x, y) which are entered in "free format" and separated by comma or blank (cf. Section 7.4). The x-values must come in increasing order, but need not be equidistant. Comment lines in tables are not acceptable, nor are blank lines. (You can, however, use `!` in the numerical records, for example `200 1.00e03 ! first entry` .) Each table terminates when the header of the next table is encountered, or when the `$ TABULAR DATA` block ends, either by encountering another block or an `$ END DATA` record. A blank line can not be used to terminate a table. When computing a y-value between table points, CHEMSIMUL uses cubic spline interpolation in x between the table entries (Section 13.6). An error stop occurs if x goes outside the table limits during the simulation.

In Section 7.14.4 we shall meet an exceptional example of an xyz table. Here each record is a triple (x, y, z), where x is the independent variable, while y and z are dependent variables. Otherwise, the xyz table is completely analogous to the xy table.

7.14.2 Tables of absolute temperature versus reaction rates

Apart from using `EA` and `B` to determine the temperature dependence of reaction rates as described in Section 7.4, it is also possible to specify an arbitrary variation by means of tables. This may be done for any reaction. In that case you should just enter `A=TABLE` in the reaction equation instead of writing `A=` ⟨some value⟩, and place the corresponding table in a `$ TABULAR DATA` block. Here is an example:

```
...
re1: OH+OH=P; a=table, q=1.0e4
...
$tabular data
```

```
table temp re1
300   1.695935e10
350   1.948324e10
400   2.197121e10
450   2.442819e10
500   2.685796e10
550   2.926351e10
600   3.164728e10
650   3.401121e10
700   3.635702e10
...
```

The header record for the table must contain `TABLE TEMP RE` followed by the identification number for the actual reaction. Each table record is a pair of values $(x, y) = (T, k)$ of absolute temperature and reaction rate, respectively.

There may be several such tables in the block, associated with different reaction equations.

7.14.3 Table of time versus temperature

You should observe that the absolute temperature T occurs in several different contexts in CHEMSIMUL. As mentioned in Section 7.15, the command `TEMP =` T defines the temperature of the reaction system in degrees Kelvin. By default $T = 298.15$ (25°C) is assumed. `TEMP` may be required in the modified Arrhenius expression (5). It can also be used in CHEMSIMUL regular expressions, i.e. plot expressions, exchange equations, and refreshable parameter definitions. In case of a thermodynamical simulation `TEMP =` T means the initial temperature.

In these cases T is either fixed or is supposed to vary during the simulation as a consequence of heat production. CHEMSIMUL admits, however, quite another way of working with the temperature: It is possible to *impose* a certain functional variation $T(t)$ of the absolute temperature T with time t, such that $T(t)$ can be used in the same way as above. This is done by placing an interpolation table in a `$ TABULAR DATA` block in the input file. Here is an example:

```
$tabular data
...
table time temp
0  298.15
1  298.50
10 300.00
1e2   305.00
3.4e3 310.00
3.0e4 320.00
3.0e5 330.00
3.0e6 340.00
3.1e7 350.00
...
$miscellaneous data
table time temp
...
```

A command `TABLE TIME TEMP` (or `TABLETIMETEMP`) in the `$ MISCELLANEOUS` block (Section 7.15) must be given to activate the table; if a table is present but no such command

is given, the table is ignored. The header record for the table must be identical with the command: `TABLE TIME TEMP`.

Each table record is a pair of values $(x, y) = (t, T)$ of time in s and absolute temperature in K, respectively. The first given time must be $t = 0$ (or, more generally $t =$ `TSTART`), and the last given time must be $\geq$ `TEND` (Section 7.6).

Clearly, the use of an imposed function $T(t)$ by such a table is not compatible with the use of the command `TEMP =` T mentioned in Section 7.15. When CHEMSIMUL discovers such a conflict, it prints a diagnostic message and aborts the simulation.

If the table function $T(t)$ exhibits a very rapid variation with time, it may be advisable to restrict the integration step length by using the `HMAX` command in order to monitor all the details in the result, cf. Section 7.6.

7.14.4 Table of time versus two temperatures

CHEMSIMUL supports a variant of the previous table, but with time versus two temperatures, as the following example shows:

```
$tabular data
...
table time temp temp2
0  298.15 15.2
1  298.50 18.1
10 300.00 22.4
1e2   305.00 26.9
3.4e3 310.00 33.2
3.0e4 320.00 45.6
3.0e5 330.00 55.7
3.0e6 340.00 60.1
3.1e7 350.00 63.5
...
$miscellaneous data
table time temp
...
```

In this table each record is a triple of values $(x, y, z) = (t, T, T_2)$, where t and T are exactly as before, while T_2 is supposed to be a "dual temperature", e.g. the temperature in another compartment of the system. CHEMSIMUL associates with T_2 the reserved name `TEMP2` (in upper case), cf. Section 6.3, and $T_2 =$ `TEMP2` can be used in all CHEMSIMUL regular expressions.

Since the program does not make any a priori assumptions about `TEMP2`, you may for instance let it represent a temperature in centigrades (or even something quite different from temperature).

You should still use the command `TABLE TIME TEMP` in the `$ MISCELLANEOUS` block (although `TABLE TIME TEMP TEMP2` will also work). For the table itself the header `table time temp temp2` is required. If `table time temp` is used as the header for such an xyz table, it will be treated as an xy table.

There can only be one table of time versus temperature in the `$ TABULAR DATA` block.

If `TEMP2` is not defined by an interpolation table, its value defaults to zero.

7.15 The Miscellaneous Block

The `$ MISCELLANEOUS` block is a "mixed bag" of items that would not fit naturally elsewhere: `CONVERT`, `DOSEUNIT`, `HCV`, `IONIC`, `MAXCON`, `R`, `TABLE TIME TEMP`, `TEMP`, `UNBAL`.

Example:

```
$Miscellaneous
convert = 1.036427e-7
doseunit = krad
hcv(H2O) = 0.018
ionic
maxcon(H[+])
r = 1.987207e-3
temp = 455
unbal
```

We give below an explanation of these various commands (alphabetic order):

CONVERT

Sets the conversion factor c (Section 2.2 and Section 2.10), if you need another value than the default value $c = 1.036427 \cdot 10^{-7}$. Recall that the conversion factor depends on the units for concentration, primary yield, and dose. For concentration given in $mol \times dm^{-3}$, primary yield given in $molecules/heV$ and (almost) pure water at 4°C, c is given in Table 4 for dose in *Gy* and *krad*, respectively. The second line in the table contains corresponding command names.

Irradiation dose	Conversion factor
`TOTALDOSE`	`CONVERT`
Gy	`1.036427E-7` (default)
krad	`1.036427E-6`

Table 4: Values of the conversion factor c

In the standard situation with *Gy* as the dose unit you need not use this command; c will be computed automatically by CHEMSIMUL. And if you are working with *krad* as the dose unit, you may just use the command `DOSEUNIT`, see below.

The `CONVERT` command may be necessary in gas kinetics where concentration is given in $molecules \times cm^{-3}$.

Another way of using `CONVERT` is to associate it with a refreshable parameter (RP). This enables the user to work with a value of c that varies during the simulation. For example you may write

```
convert = <cvar
```

The RP `cvar` should be defined in the Refreshable Parameters Block, Section 7.11.

DOSEUNIT

Selects either *Gy* or *krad* as unit for the dose. The choice determines the value of the conversion factor c according to Table 4 above. You may write

```
doseunit = Gy
```

or

```
doseunit = krad
```

Only the leading letter on the right-hand side is significant, and the unit designator is case-insignificant. By default *Gy* is assumed.

HCV

In Section 2.7 it was discussed how CHEMSIMUL can deal with adiabatic processes. When simulating such a system, the specific heat capacity `HCV` $= c_v(\mathrm{R}_s)$ should be entered for the species involved, as well as the heat of reaction, `Q` $= q_r$ (Section 7.4), for the reactions. The units for c_v and q_r must be compatible, cf. Table 1 in Section 2.1. By default the specific heat capacity values are set to zero.

Hint: In diluted aqueous solutions it is appropriate to enter `HCV` *for water.*

IONIC

This command enables the automatic computation of a number of auxiliary quantities, which may be useful for calculations with ionic strength, cf. Section 10.3. By default this feature is disabled.

MAXCON

To find the maximum concentration value $[\mathtt{X}]_{\max}$ for a species `X` (Section 5.3), you should type the command `MAXCON(X)`. If you want the maximum concentration of more species, these should be entered as a comma-separated list, e.g. `MAXCON(X,Y,Z)`. Note that only one `MAXCON` command is allowed.

R

The gas constant R in the units of your choice. This command must be used if you use the Arrhenius correction of the reaction constants (Section 2.6). You must ensure that the units of R and activation energy E_a are compatible, see also Table 1 in Section 2.1. By default $R = 8.314472$ is assumed, corresponding to the unit $J \times K^{-1} \times mol^{-1}$.

TABLE TIME TEMP

(or `TABLETIMETEMP`). This command must be used to activate a table of time versus temperature as explained in Section 7.14.3.

TEMP

The command `TEMP =` T defines the temperature T of the reaction system in K. By default $T = 298.15$ (25°C). In a thermodynamic simulation, where the temperature is supposed to vary, `TEMP` means the initial value of the temperature.

UNBAL

The command `UNBAL` enables the user to overrule the halting of a simulation when a stoichiometric unbalance has been detected. The check itself can never be switched off. WARNING: This facility should only be applied in exceptional cases; reliable simulation results are only guaranteed when the mass balance is fulfilled.

8 Using exchange equations

In this chapter we give some examples of using exchange equations in CHEMSIMUL, cf. Section 7.10. Exchange equations are a supplement to the kinetics equations in the system. They can model a great variety of physical or physico-chemical processes. The only requirement is that the process can be described by linear ordinary differential equations. Zero-order reactions are just a special case of exchange equations, cf. Section 11.7.

We have already seen an example of exchange equations for modelling convection processes in Section 7.10 As another example, consider the diffusion of aqueous hydrogen from the liquid surface to the bulk in one-dimensional geometry. This process can be modelled by the first-order differential equations

$$\frac{\mathrm{d}[\mathrm{H}_2]}{\mathrm{d}t} = -\frac{3D}{d^2}\Big([\mathrm{H}_2] - [\mathrm{H}_{2s}]\Big) \tag{30}$$

and

$$\frac{\mathrm{d}[\mathrm{H}_{2s}]}{\mathrm{d}t} = \frac{3D}{d^2}\Big([\mathrm{H}_2] - [\mathrm{H}_{2s}]\Big) \tag{31}$$

where D here is the diffusion coefficient, d is the depth of liquid, H_2 the bulk hydrogen and H_{2s} the surface hydrogen.

The corresponding write-up of these equations within an exchange equation block would be similar to the example given in Section 7.10.

Likewise, other heterogenous phenomena as precipitation and gas transport can be modelled by exchange equations.

Next we show an example of a quite different character, which illustrates the power and versatility of the exchange equations. It is taken from Forsythe *et al.* [11] and contains no chemistry at all. The problem is from dynamics and concerns the motion of two bodies under mutual gravitational attraction. Newton's laws of motion provide two second-order differential equations which are expanded into a system of 4 first-order equations. These in turn are formulated in terms of exchange equations as shown. Note that initial values are given by `CON` statements.

```
$identification
Motion of two bodies under mutual gravitational attraction.
$exchange
d(y1)/dt = y3
d(y2)/dt = y4
d(y3)/dt = -y1/R
d(y4)/dt = -y2/R
$symbolic
#alpha = 0.7853982 ! pi/4
$refresh
<R = sqrt(y1^2+y2^2)^3/alpha^2
$concentrations
```

```
CON(y1) = 0.75 ! 1 - e (e=eccentricity) with e=0.25
CON(y4) = 1.013945 ! alpha*sqrt((1+e)/(1-e)) with alpha=pi/4
$integration
TEND=12
$graphics
pe1:y1
pe2:y2
plot(pe1;pe2)
$output control
prints = 24
$END DATA
```

This example also shows the use of a refreshable parameter `R` as a means of storing a current account.

9 Radiolysis from decaying isotopes

CHEMSIMUL has a module for simulating the chemical radiolysis accompanying the storage of radioactive waste. In particular, we shall describe how isotopic data are entered into the `$ ISOTOPE` block in the input file, cf. Section 7.12.

If you have no need for solving problems of this kind, you may skip the entire chapter.

Note that using the `$ ISOTOPE` block together with the `$ IRRADIATION` block will result in a warning message from CHEMSIMUL.

9.1 Scope

We are interested in computing effects from a mixture of exponentially decaying radioactive nucleides, which may be organized in isotope families. Such effects include dose rates and primary production of chemical species. These calculations are done analytically by CHEMSIMUL. The mathematical methods, which are based on Laplace transforms and graph theory, are detailed in Sections 13.8 and 13.10.

The nuclear reactions proceed independently of the chemical system. But the radiation from the isotopes may conversely affect the chemistry through primary production (or destruction) of chemical species. (Indeed, it is also possible to ignore this and just use the isotopic part of CHEMSIMUL alone without doing any chemical simulation at all.)

The reaction scheme for decaying isotopes is similar to a chemical reaction scheme with first order reactions.

9.2 Fundamental concepts

Suppose that one of the nuclear decay reactions is

$$p \rightarrow q \tag{32}$$

which means that the mother isotope p decays into some daughter isotope q. The rate constant k for this process is connected to the period (half-life) τ by the formula

$$k = \frac{\ln 2}{\tau} \tag{33}$$

Let us first assume that our nuclear system comprises only the two isotopes p and q in (32). Then it is well-known that the number of atoms $N(t)$ of p, contained at time t in some specific "system volume", obeys the differential equation

$$\frac{\mathrm{d}N(t)}{\mathrm{d}t} = -kN(t) \tag{34}$$

and hence decays exponentially according to the formula

$$N(t) = N(0)e^{-kt} \tag{35}$$

where k is the rate constant from (33). We call $N(t)$ the *amount* of atoms of the nuclear species p. This quantity is analogous to the concentration $[\mathrm{R}](t)$ of a chemical reactant R in a first-order reaction. We also define the *activity* $A(t)$ of the reaction by

$$A(t) = kN(t) \tag{36}$$

We observe from (34) that $A(t)$ is simply the rate of disintegration of p. In particular, (36) gives for $t = 0$ the *initial activity*

$$A(0) = kN(0) \tag{37}$$

Amount may be measured in number of atoms in the system. Activity is usually measured in becquerel (Bq), where 1 Bq corresponds to 1 nuclear decay per s.

9.3 Isotope filiation as first-order systems

In (32) the daughter product q may be a stable isotope. Often, however, we have a chain of nucleides $p_1, p_2, \ldots, p_r, p_*$, where the mother isotope p_1 decays to p_2, p_2 decays to p_3, etc., and the chain terminates when p_r decays to a stable isotope p_*. This is called a *linear isotope family* of length r. For $r = 1$ the family degenerates to a single decaying isotope. There may be more than one isotope family in the system considered.

As a simple example, consider a linear isotope family of length 2,

$$p \to q \to r \tag{38}$$

with rate constants k_p, k_q, k_r ($k_r = 0$). In Section 13.8 we shall discuss the governing differential equations and how they are solved; here we just list the resulting expressions for the amounts $N_p(t)$, $N_q(t)$, and $N_r(t)$:

$$N_p(t) = N_p(0)e^{-k_p t} \tag{39}$$

$$N_q(t) = N_p(0)\frac{k_p}{k_q - k_p}\left(e^{-k_p t} - e^{-k_q t}\right) + N_q(0)e^{-k_q t} \tag{40}$$

$$N_r(t) = N_p(0)\left(1 - \frac{k_q}{k_q - k_p}e^{-k_p t} + \frac{k_p}{k_q - k_p}e^{-k_q t}\right) + N_q(0)(1 - e^{-k_q t}) + N_r(0) \tag{41}$$

It is easy to check that the initial amounts are correct and that the sum condition

$$N_p(t) + N_q(t) + N_r(t) = N_p(0) + N_q(0) + N_r(0) \tag{42}$$

is fulfilled. Activities can now be computed from (39–40) by (36). We see that (40–41) break down when $k_p = k_q$. However, coinciding rate constants can be dealt with in an efficient and elegant way by the Laplace transform method as explained in Section 13.8.

Families may be more complex than this example, as *branching* may occur, which means that there may be more than one decay mode for an isotope somewhere in the chain. Thus the same isotope may be involved in more than one decay process. An example is given in Section 9.5.

To cope with such complications we resolved that CHEMSIMUL should be able to handle any isotopic decay scheme that can be expressed by sets of nuclear reactions (32), the only restriction being that no isotope may, directly or indirectly, decay into itself.

Thus our nuclear decay model will simply be a general system of *first-order reactions*. Such a nuclear system is quite similar to a chemical system of first order reactions; this is also reflected in the input format, cf. Section 9.5.1.

Each nuclear reaction j is like (32) and identifies a mother isotope $i = i(j)$ $(i = 1, \ldots, m)$. However, several reactions may have the same mother isotope. This is because some isotopes can occur as mother isotopes in more than one reaction in case of branching, as in Figure 2 in Section 9.5.2.

CHEMSIMUL is able to decode the filiation structure from any set of nuclear reactions by the methods described in Section 13.8, and then compute analytical expressions for the current amounts $N_i(t)$ and activities $A_i(t)$ of all decaying isotopes. This analytical approach is in contrast to the numerical solution procedure used by CHEMSIMUL for the chemical reaction system.

9.4 Isotopic radiolysis

Each decay process (32) will be accompanied by α, β, γ, or neutron radiation, which may create chemical species. When a mixture of m decaying isotopes is present, we assume that we can express the production rate of the species R_s $(s = 1, \ldots, N)$ in terms of dose rates by the following formula:

$$\frac{\mathrm{d}[\mathrm{R}_s]}{\mathrm{d}t} = c\Big(G_s^{\alpha} D'_{\alpha}(t) + G_s^{\beta} D'_{\beta}(t) + G_s^{\gamma} D'_{\gamma}(t) + G_s^{\mathrm{n}} D'_{\mathrm{n}}(t)\Big) \tag{43}$$

Here $G_s^{\alpha}, G_s^{\beta}, G_s^{\gamma}, G_s^{\mathrm{n}}$ are primary yields ("G-values") for radiolytic production of R_s by α, β, γ, and neutron radiation, respectively, while $D'_{\alpha}(t)$, etc., are corresponding dose rates. Finally c is the conversion factor introduced in Section 2.2. (Note the similarity of (43) and (1) in Section 2.2.)

Let us write $D'_{\alpha}(t)$, $D'_{\beta}(t)$, $D'_{\gamma}(t)$, $D'_{\mathrm{n}}(t)$ as sums of contributions from the nuclear reactions. For example, for the α dose rate we have

$$D'_{\alpha}(t) = \sum_j d'_{\alpha j}(t) \tag{44}$$

The term $d'_{\alpha j}(t)$ is called the *specific dose rate* for α radiation in reaction j. The dose rates $D'_{\beta}(t)$, $D'_{\gamma}(t)$, $D'_{\mathrm{n}}(t)$ are evaluated in the same way.

As mentioned, CHEMSIMUL computes analytical expressions for the activities $A_i(t)$, $i = 1, \ldots, m$. This computation requires the knowledge of the activities at a given time t_0 which for convenience will be taken to time zero.

The specific dose rates $d'_{\alpha j}(t)$, $d'_{\beta j}(t)$, $d'_{\gamma j}(t)$, $d'_{\mathrm{n}j}(t)$, are at any time proportional to the activity $A_i(t)$, where $i = i(j)$ is the mother isotope in reaction j:

$$\frac{d'_{\alpha j}(t)}{d'_{\alpha j}(0)} = \frac{d'_{\beta j}(t)}{d'_{\beta j}(0)} = \frac{d'_{\gamma j}(t)}{d'_{\gamma j}(0)} = \frac{d'_{\mathrm{n}j}(t)}{d'_{\mathrm{n}j}(0)} = \frac{A_i(t)}{A_i(0)} \tag{45}$$

(provided the denominators are > 0). This means that we can evaluate say $d'_{\alpha j}(t)$ by

$$d'_{\alpha j}(t) = d'_{\alpha j}(0)\frac{A_i(t)}{A_i(0)} = d'_{\alpha j}(0)\frac{N_i(t)}{N_i(0)} \tag{46}$$

where we have used the proportionality between activity and amount expressed in (36).

Thus, to evaluate (44) we should just supply the relevant initial values $d'_{\alpha j}(0)$, etc. To summarize, each nuclear reaction (32) is characterized by the following parameters:

$$p \to q; \quad k, \quad d'_{\alpha},\ d'_{\beta},\ d'_{\gamma},\ d'_{\mathrm{n}} \tag{47}$$

where $d'_{\alpha} = d'_{\alpha j}(0)$ etc.

When CHEMSIMUL sets up the differential equations, the sum in (43) will be added to the right-hand side (cf. (23), (26), and (88)).

9.5 Isotope input and output: an example

This section gives prescriptions for making proper input for radiolysis from isotopes.

All isotope data must be contained in the `$ ISOTOPE` data block in the CHEMSIMUL data file, see Chapter 7 for rules of data blocks. The block ends with the next CHEMSIMUL data block headed by a `$` record, or with the `$ENDDATA` record.

The data within the Isotope Block may come in any order, except for commands giving selective dose-rate printing, see later.

As for other blocks, comment lines and blank lines are accepted within the Isotope Block.

9.5.1 Sample input

In Section 7.12 we listed the `$ ISOTOPE` block for a sample case, which comes from a realistic test data set called `ISOEXAMP`. (The complete data file is published on the CHEMSIMUL website `www.chemsimul.dk`.)

We note the similarity in the write-up of the nuclear reaction system with a chemical reaction system. There is one important difference, however: For the chemical simulation initial values are assigned to *concentrations* which are analogous to *amounts* of atoms in the nuclear case. For the latter we find it more natural to state initial values for *activities*. (The activity and amount formulations are indeed equivalent, except for the usually irrelevant possibility of assessing amounts of stable isotopes.)

9.5.2 Filiation structure in the example

The `ISOEXAMP` case is interesting since it has two isotope families as Figure 2 shows:

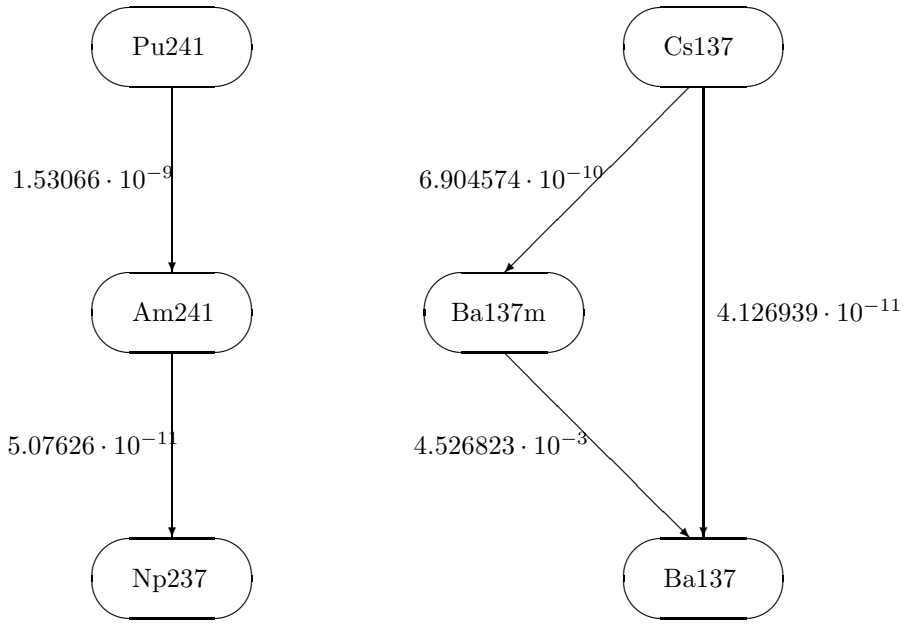

Figure 2. Linear isotope family and family with branch

One family is linear, while the other one has a branch. For each decay the corresponding rate constant is shown. We shall see how the `$ ISOTOPE` block in Section 7.12 encodes this filiation structure.

9.5.3 Rules for preparing isotope input

In general the input for the nuclear reactions is given in a way that conforms with (47). Each nuclear reaction equation is prefixed with a unique identification `NR`n:, where `NR` means `N`uclear `R`eaction and n is the equation identification number. (This syntax is similar to that for the chemical reactions.) Then follows the name chosen for the mother isotope, the character `=`, and then the name chosen for the daughter isotope followed by a semicolon. Note that isotope names can be chosen freely as alphanumeric text strings; they may begin with digits, as `243Cm`. These names are case sensitive, but otherwise all commands or specifiers in the Isotope Block are case insensitive.

In the nuclear reaction equation the reaction rate is defined by `k=`, while the initial dose rates for α, β, γ, and neutron irradiation are defined by the 4 terms: `DA = ...`, `DB = ...`, `DG = ...`, and `DN = ...`. Thus the specifiers `k=`, `DA=`, `DB=`, `DG=`, `DN=` correspond to k, d'_α, d'_β, d'_γ, and d'_n in (47). Omitted specifiers are interpreted as zero. The internal order of the specifiers is immaterial.

The `ACTIVITY` command specifies initial activities for the isotopes; the syntax is clear from the example.

You should notice that several reactions may contribute to the activity of the same isotope. This will happen when branching occurs. In any case you should always state the *total* initial activity for each isotope.

Use of the specifiers `DA=`, `DB=`, `DG=`, or `DN=`, requires an initial activity for the mother isotope. Activities must not be specified for stable isotopes.

Inputting of the nuclear G-values in (43) is straightforward: We write `GA(X)`, `GB(X)`, `GG(X)`, `GN(X)` for the nonzero primary yields G_s^α, G_s^β, G_s^γ, G_s^n, for producing a given species X = X$_s$. See the sample input in Section 7.12. Negative primary yields correspond to destruction of species. Refreshable parameter input is supported with the same syntax as indicated in Section 7.8 for the radiolytic primary yields. For example you may write `GG(H2) = <ggh2` and define `ggh2` in your RP block.

It is possible to supplement the standard overall dose-rate tables with additional tables with specific dose-rate contributions from given radiation types (`A`, `B`, `G`, `N`), and selected isotopes. Such a table is generated whenever one or more of the commands `DOSERATEA`, `DOSERATEB`, `DOSERATEG`, `DOSERATEN`, are present. Each of them takes a comma-separated list of isotope names as argument. In our example we have two such commands.

We also mention that there is an isotope command `DECAYPRINT` which causes tables to be printed of the decaying amounts and activities of all the isotopes. For isotopes with no initial activity the initial amounts are (arbitrarily) set to zero. This feature may be useful for test purposes. (It was not used in the present example.)

Note that nuclear G-values can enter *regular expressions* (Sections 6.4) as a *braced item* with the same syntax as for the radiolytic G-values, say {`GG(H2)`}. The same RP restriction applies here too, i.e. if you have defined `GG(H2) = <ggh2` you cannot write {`GG(H2)`} in an expression but should instead use `ggh2`.

Moreover, the 4 kinds of total doserates by emission type can occur in regular expressions, by writing {`DOSERATEA`}, {`DOSERATEB`}, {`DOSERATEG`}, or {`DOSERATEN`}. Such an item will contain the value of the specific kind of doserate, summed over isotopes, at the actual time

t of the simulation.

9.5.4 Sample output

Here we show the isotope part of the CHEMSIMUL result file for our sample case `ISOEXAMP`. (To save space not all the table contents are shown here.) We begin with the input-echo part:

```
================== INPUT DATA: ISOTOPES ==================

NUCLEAR REACTION SYSTEM
-----------------------
NR1:Pu241=Am241;k=1.53066E-9,DB=2.80E-11
NR2:Am241=Np237;k=5.07626E-11,DA=4.12E-5,DG=1.70E-9
NR3:Cs137=Ba137m;k=6.904574E-10,DB=6.29E-5
NR4:Cs137=Ba137;k=4.126939E-11,DB=8.98E-6
NR5:Ba137m=Ba137;k=4.526823E-3,DG=2.15E-4

INITIAL ACTIVITIES
------------------
ACTIVITY(Pu241)=5.2E12
ACTIVITY(Am241)=5.0E10
ACTIVITY(Cs137)=1.755E13
ACTIVITY(Ba137m)=1.755E13

PRIMARY YIELDS
--------------
SPECIES                      A           B           G           N
E[-]                      0.060000    2.660000    2.660000
OH[-]                                 0.100000    0.100000
OH                        0.240000    2.670000    2.670000
H2O                      -2.710000   -6.970000   -6.970000
H2O2                      0.985000    0.720000    0.720000
HO2                       0.220000
H                         0.210000    0.550000    0.550000
H2                        1.300000    0.450000    0.450000
H[+]                      0.060000    2.760000    2.760000

NO. OF NUCLEAR REACTIONS         =          5
NO. OF INITIAL ACTIVITIES        =          4
NO. OF ISOTOPES                  =          6
```

In the result file you will find two standard tables of accumulated doses as well as dose rates, originating from the decaying isotopes. These tables look as follows:

```
================ OUTPUT TABLES: ISOTOPES ================

                ACCUMULATED DOSES FROM ISOTOPES
-----------------------------------------------------------------------
    TIME         A           B           G           N         TOTAL
 0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
 3.1558E+10  2.8640E+06  9.8233E+04  2.7737E+05  0.0000E+00  3.2396E+06
 6.3115E+10  3.4604E+06  9.8233E+04  2.7740E+05  0.0000E+00  3.8360E+06
```

```
 ...
 2.5246E+11  3.6109E+06  9.8233E+04  2.7740E+05  0.0000E+00  3.9866E+06
 2.8402E+11  3.6109E+06  9.8233E+04  2.7740E+05  0.0000E+00  3.9866E+06
 3.1558E+11  3.6109E+06  9.8233E+04  2.7740E+05  0.0000E+00  3.9866E+06

                      DOSE RATES FROM ISOTOPES
-------------------------------------------------------------------------
    TIME         A           B           G           N         TOTAL
 0.0000E+00  4.1200E-05  7.1880E-05  2.1500E-04  0.0000E+00  3.2808E-04
 3.1558E+10  3.7918E-05  6.7310E-15  1.5646E-09  0.0000E+00  3.7920E-05
 6.3115E+10  7.6406E-06  6.3030E-25  3.1527E-10  0.0000E+00  7.6409E-06
 ...
 2.5246E+11  5.1148E-10  4.2499E-85  2.1105E-14  0.0000E+00  5.1150E-10
 2.8402E+11  1.0307E-10  3.9797E-95  4.2527E-15  0.0000E+00  1.0307E-10
 3.1558E+11  2.0768E-11  0.0000E+00  8.5694E-16  0.0000E+00  2.0769E-11
```

Because it was requested in the input, there will also appear a selective dose-rate table:

```
SELECTIVE DOSE RATES BY TYPE AND ISOTOPE
----------------------------------------
     TIME       DB(Pu241)    DA(Am241)
   0.0000E+00   2.8000E-11   4.1200E-05
   3.1558E+10   2.9445E-32   3.7918E-05
   6.3115E+10   3.0965E-53   7.6406E-06
   ...
   2.5246E+11   0.0000E+00   5.1148E-10
   2.8402E+11   0.0000E+00   1.0307E-10
   3.1558E+11   0.0000E+00   2.0768E-11
```

9.5.5 Saving isotope results for manual restart

As outlined in Section 7.15 it is possible to store chemical concentration values at $t = t_{\text{end}}$ in a save file, by using the `SAVE` command. This save facility extends to isotopic data whenever an Isotope Block is present, thus admitting an easy way to make "manual restarts" also when isotopes are involved. To ensure data integrity and easy copying, CHEMSIMUL inserts a complete revised Isotope Block at the end of the save file, as the small example below shows. We suppose that a data file contains `SAVE` and has the following Isotope Block:

```
$ISOTOPES
NR1: I1 = XXX; K=0.00693, DA=1, DG=1 ! Period = 100
NR2: I2 = XXX; K=0.00347, DA=1, DG=1 ! Period = 200
GG(OH)=2.67
...
GA(H2O)=-5.46
ACTIVITY(I1)=0.00693
ACTIVITY(I2)=0.00347
```

Then, at the end of simulation, a save file with a revised Isotope Block is created:

```
...
CON(MH2OH2)  =  2.1591181174106500E-05
***
*** ISOTOPE BLOCK INTENDED TO REPLACE PREVIOUS BLOCK BEFORE RESTART:
```

```
$ISOTOPES
NR1:I1=XXX; K=0.00693, DA= 5.00073596E-01, DG= 5.00073596E-01
NR2:I2=XXX; K=0.00347, DA= 7.06805328E-01, DG= 7.06805328E-01
GG(OH)=2.67
...
GA(H2O)=-5.46
ACTIVITY(I1) = 3.46551002E-03
ACTIVITY(I2) = 2.45261449E-03
***
*** ENDSAVE ... END OF SAVE FILE
```

By comparing this with the input file you will notice that the dose coefficients `DA=`... etc., as well as the activities are replaced by end simulation values, while the remaining Isotope Block is left intact. You may then copy the block as a whole into an input file and let it replace the Isotope Block there.

10 Ionic strength calculations

The *ionic strength* is a parameter which characterizes the ionic contents of aqueous solutions. As a consequence of ionic strength, rate constants between two charged reactants may be altered. This *salt effect* is important in the theory of electrolytes. CHEMSIMUL has no direct procedures for computing the quantities mentioned in this chapter, and it has no particular data block for ionic strength. Instead the program offers a number of tools for easing your simulation work when this involves ionic strength. Refreshable parameters (RPs) play an important role in such calculations.

10.1 Ionic strength definitions

First we give the definition of ionic strength based on conventional volumetric units:

$$I = \frac{1}{2}\sum_i C_{\mathrm{X}_i} z_{\mathrm{X}_i}^2 \tag{48}$$

where C_{X_i} is the molarity concentration (mol/dm^3 solution) of ion X_i, z_{X_i} is the charge of that ion (positive or negative), and the sum is taken over all ions in the solution. For a 1:1 electrolyte such as sodium chloride, the ionic strength is equal to the concentration. For multivalent ions, the ionic strength is greater than the concentration. I is measured in mol/dm^3.

However, for a more accurate estimation of the salt effect it is better to replace (48) with the following formula:

$$I_* = \frac{1}{2}\sum_i A_{\mathrm{X}_i} z_{\mathrm{X}_i}^2 \tag{49}$$

This differs from (48) in the term A_{X_i}, which is the molality concentration (mol/kg solvent) of ion X_i, cf. Section 2.9. (To discern the two, I is sometimes called the pseudo ionic strength and I_* the true ionic strength.) If we substitute (14) from Section 2.9 in (49) we obtain the simple conversion formula

$$I_* = \frac{I}{\mathrm{CUC}} \tag{50}$$

where CUC was defined in (13) in Section 2.9. I_* is measured in mol/kg.

10.2 Activity and Davies' formula

Activity is a concept involved in various problems such as pH calculation, chemical equilibria, solubility, etc. The specific definitions of activity and activity coefficients depend on context. Here we shall just give a couple of examples. First, the activity of water is given by the formula

$$\mathrm{Act}_{\mathrm{H2O}} = \frac{C_{\mathrm{H2O}}}{C_{\mathrm{H2O}} + \sum C_{\mathrm{X}_i}} \tag{51}$$

where C_{H2O} is the concentration of water, while C_{X_i} is the concentration of solute X_i, both given as molarities. Next we shall give the formula for an *activity coefficient* γ, which is the main application of ionic strength. For a species X with charge z_{X} this can be calculated from:

$$\log_{10}\gamma = -0.5092 \times z_{\mathrm{X}}^2 \left(\frac{\sqrt{I_*}}{1+\sqrt{I_*}} - 0.3 I_* \right) \tag{52}$$

This formula is based on the Debye-Hückel model with the modification of Davies who proposed the correction $-0.3I_*$. It is therefore called *Davies' formula*.

An important kinetic application of (52) is to estimate the rate constant k under influence of the salt effect. The resulting expression becomes

$$\log_{10}\left(\frac{k}{k_0}\right) = 1.0184\, z_A z_B \left(\frac{\sqrt{I_*}}{1+\sqrt{I_*}} - 0.3 I_* \right) \tag{53}$$

or, equivalently,

$$k = k_0 \times 10^{1.0184\, z_A z_B \ (\sqrt{I_*}/(1+\sqrt{I_*}) - 0.3 I_*)} \tag{54}$$

Here k is the rate constant at the ionic strength I_*, and k_0 the rate constant at the ionic strength zero. The numbers z_A and z_B are the charges of the two reactants. For example, we have $z_A = -1$ and $z_B = 2$ in the following reaction:

$$\mathrm{S}^- + \mathrm{T}^{++} = \mathrm{S} + \mathrm{T}^+ \tag{55}$$

The expression (53) corresponds to the original Brønsted-Bjerrum relation for the salt effect, except for Davies' correction $-0.3I_*$ [12, 13].

When ionic strength is applied to compute activity or kinetic corrections, as in (52) and (53), the maximum ionic strength compatible with the model is about 0.5 *mol/kg*; in more general applications this limit will be about 0.3 *mol/kg*.

10.3 Ionic strength with CHEMSIMUL

We shall now describe how the quantities in Sections 10.1 and 10.2 can be calculated in CHEMSIMUL by means of predefined refreshable parameters (RP). These include `ION` for I in (48), `SIGMACi` for $\sum_i C_{\mathrm{X}_i}$ in (51), and `SIGMAMiCi` for $\sum_i M_{\mathrm{X}_i} C_{\mathrm{X}_i}$ in (13) in Section 2.9. Note that the sums `SIGMACi` and `SIGMAMiCi` exclude the solvent $\mathrm{H_2O}$. To ensure that CHEMSIMUL makes this exclusion properly, you must use the species name `H2O` (upper case) for water.

Observe also that CHEMSIMUL extracts the charges z_{X_i} of species X_i from their names, in order to compute I in (48). You must therefore provide proper square bracket qualifiers of the species names, e.g. `CO3[--]`, cf. the naming rules for CHEMSIMUL species given in Section 6.3.

We give below an example to explain the use of the ionic strength features, where we show an excerpt of the data file for a simulation case with carbonation of pure water. We begin by listing part of the `$ CHEMICAL REACTIONS` block:

```
$ CHEMICAL REACTIONS
* Water equilibrium H2O/OH[-], pKe = 13,9953
RE67:H2O+H2O=OH[-]+H3O[+];A=<k67
RE68:OH[-]+H3O[+]=H2O+H2O;A=1.103078125E11
...
```

Besides water the total system contains 6 species called `OH[-]`, `H3O[+]`, `HCO3[-]`, `CO3[--]`, `CO2`, and `H2CO3`. In order to activate the ionic strength facilities you should put the command `IONIC` in the `$ MISCELLANEOUS` block:

```
$ MISCELLANEOUS
...
ionic
```

Without this command CHEMSIMUL will not establish the predefined parameters mentioned above. In the `$ SYMBOLIC CONSTANTS` block you put the molar masses (kg/mol) for all the solute species as follows (including a couple of other constants):

```
$ SYMBOLIC CONSTANTS
...
* Molar Masses (kg/mol)
# MMOH[-]=17.00734E-3
# MMH3O[+]=19.02322E-3
# MMHCO3[-]=61.01714E-3
# MMCO3[--]=60.0092E-3
# MMCO2=44.0098E-3
# MMH2CO3=62.02508E-3
...
# RHOw=996.999 ! Density of water (kg/m3)
# Kw=1.0108470836822549E-14 ! Water dissociation constant
...
```

The full species name should be immediately preceded by `MM` (upper case). As a result, CHEMSIMUL generates an *internal RP* called `SIGMAMiCi`. This can be used in the same way as the "normal" RPs defined in the `$ REFRESHABLE PARAMETERS` block, i.e. it works exactly as an RP defined by

```
<SIGMAMiCi = MMOH[-] * OH[-] + ... + MMH2CO3 * H2CO3
```

CHEMSIMUL will not generate `SIGMAMiCi` if some molar masses are missing, or none are present at all, but it will still generate `ION` and `SIGMACi`. This might be OK for you, if you do not intend to work with molalities. On the other hand, the presence of molar masses for species that are not in the reaction system, will do no harm; you may for example wish to have a "library" of molar masses placed in your `$ SYMBOLIC CONSTANTS` block.

Proceeding now to the `$ REFRESHABLE PARAMETERS` block, we see several essential simulation quantities being formulated here:

```
$ REFRESHABLE PARAMETERS
* RPs SIGMAMiCi, SIGMACi, ION automatically generated by CHEMSIMUL
< RHOLIQ=RHOw/2+104.673*SQRT(SIGMAMiCi+2.281766E-5*RHOw^2) ! Density
< CUC=1E-3*RHOLIQ-SIGMAMiCi ! Conversion of Unit of Concentration
< IS=ION/CUC ! Conversion into true ionic strength
< gamma=10^(-0.5092*(SQRT(IS)/(1+SQRT(IS))-0.3*IS)) ! Act.coeff.(Davies)
```

```
* Water equilibrium:
< ActH2O=H2O/(H2O+SIGMACi) ! Activity of water
< k67=1.103078125E11*Kw*(H2O*CUC/(gamma*ActH2O))^2 ! feedback to RE67
...
```

In summary, you must formulate your ionic strength model yourself, but CHEMSIMUL provides facilities to reduce your work with this.

11 Hints for special problems

Although many simulation problems in chemical kinetics at first sight do not appear to be directly solvable by CHEMSIMUL, a reformulation of the original problem may sometimes help. In this chapter you will find a number of hints which enable you to solve such special problems with the program.

11.1 How to mark a reaction

If you want to know how much material is passing through a single reaction you can mark the reaction without sacrificing the material balance: The reaction

$$\texttt{RE1: A + B = C + D; A} = k_1 \tag{56}$$

is marked like this:

$$\texttt{RE1: A + B = AB; A} = k_1 \tag{57}$$

$$\texttt{RE2: AB + M = C + D + N; A} = k_2 \tag{58}$$

$$\texttt{CON(M)} = M \tag{59}$$

To ensure that `RE1` is rate determining during the whole simulation you must choose high values for M and k_2:

$$M \times [\mathrm{AB}] \times k_2 \gg [\mathrm{A}] \times [\mathrm{B}] \times k_1 \tag{60}$$

The simulated concentration [N] represents the yield of this reaction.

11.2 Maintain constant concentration of solute

The program is designed to preserve the mass balance. In view of this you may use either of two possible methods, if you want to maintain a constant concentration for a solute `X`:

11.2.1 The equilibrium method

Here you assign a high initial concentration M to a `DUMMY` species in equilibrium with `X`:

$$\texttt{CON(DUMMY)} = M \tag{61}$$

$$\texttt{RE1: DUMMY = X; A} = a \tag{62}$$

$$\texttt{RE2: X = DUMMY; A} = b \tag{63}$$

Choose a value for the rate constant a (e.g. 100) and calculate b from the equation:

$$M \times a = [\mathrm{X}] \times b \tag{64}$$

M must be so high that the change in concentration during the simulation is insignificant.

11.2.2 The conservation method

This method recreates X whenever X is consumed in a reaction, and removes X whenever X is produced in a reaction. As an example consider the reactions:

$$\texttt{RE1: X} + \texttt{N} = \texttt{X}[-] + \texttt{N}[+];\ \texttt{A} = a$$
$$\texttt{RE2: C} + \texttt{B} = \texttt{X} + \texttt{D};\ \texttt{A} = b \tag{65}$$

Assign a high DUMMY concentration as in (61). Change the equations as follows:

$$\texttt{RE1: X} + \texttt{N} = \texttt{X}[-] + \texttt{XN}[+];\ \texttt{A} = a$$
$$\texttt{RE2: C} + \texttt{B} = \texttt{DUMMY} + \texttt{D};\ \texttt{A} = b \tag{66}$$

and add the reaction

$$\texttt{RE3: XN}[+] + \texttt{DUMMY} = \texttt{X} + \texttt{N}[+];\ \texttt{A} = c \tag{67}$$

Choose the rate constant c so high that X is produced much faster in RE3 than it is consumed in RE1.

11.3 Equilibrium of gas phase with solution

If you have a gas phase (volume = g) in equilibrium with a liquid phase (volume = ℓ) and you want to maintain the equilibrium of for example oxygen during the simulation, this can be accomplished in the following way:

First determine the equilibrium concentration ratio of dissolved oxygen O_2 and gas phase oxygen $O_{2,gas}$:

$$\frac{[O_2]}{[O_{2,gas}]} = r \tag{68}$$

Then introduce a DUMMY species whose concentration [DUMMY] is calculated as the oxygen concentration if the gas phase had the same volume as the liquid phase:

$$[\text{DUMMY}] = [O_2] \times \frac{g}{\ell} \tag{69}$$

Finally determine the rate constants a and b in the equilibrium reactions

$$\texttt{RE1: DUMMY} = \texttt{O2};\ \texttt{A} = a \tag{70}$$
$$\texttt{RE2: O2} = \texttt{DUMMY};\ \texttt{A} = b \tag{71}$$

by first choosing a value for a (e.g. 100) and then calculating b from the equation:

$$[\text{DUMMY}] \times a = [O_2] \times b \tag{72}$$

11.4 Mass balance for G-values

CHEMSIMUL does not check the mass balance for the G-values. The reason is that each species name is an indivisible entity with no possibility of analysing its atomic contents. (It is an important feature in CHEMSIMUL that names can be chosen freely, especially when organic molecules are involved.)

In case of water irradiation it is possible to preserve the mass balance, if you make sure that the following 3 equations are fulfilled:

1. Hydrogen balance:

$$2\texttt{G(H2O)} + \texttt{G(H)} + \texttt{G(H[+])} + \texttt{G(OH)} + \texttt{G(HO2)} + 2\texttt{G(H2)} + 2\texttt{G(H2O2)} + \texttt{G(OH[-])} = 0 \tag{73}$$

2. Oxygen balance:

$$\mathtt{G(H2O)} + \mathtt{G(OH)} + 2\mathtt{G(HO2)} + 2\mathtt{G(O2[-])} + 2\mathtt{G(H2O2)} + \mathtt{G(O)} + \mathtt{G(OH[-])} = 0 \qquad (74)$$

3. Charge balance:

$$\mathtt{G(H[+])} = \mathtt{G(e[-])} + \mathtt{G(O2[-])} + \mathtt{G(OH[-])} \qquad (75)$$

Note that `G(H2O)` is negative.

11.5 Handling of an equilibrium

Do not attempt to define an equilibrium by more than one set of equations. For example, the H_2O_2/HO_2^- equilibrium should be defined by either of the following two sets of reactions,

$$\begin{aligned} H_2O_2 + H_2O &= HO_2^- + H_3O^+ \\ HO_2^- + H_3O^+ &= H_2O_2 + H_2O \end{aligned} \qquad (76)$$

or

$$\begin{aligned} H_2O_2 + OH^- &= HO_2^- + H_2O \\ HO_2^- + H_2O &= H_2O_2 + OH^- \end{aligned} \qquad (77)$$

The use of both sets will not add to the accuracy but may slow down the integration. Use (76) in acid solutions and (77) in alkaline solutions.

11.6 Using save files for "manual restarts"

As explained in Section 7.15 the `SAVE` command can be used to store end concentration values (and Isotope Blocks when relevant) to a save file, and then this file could be used to edit a new input file for continued simulation.

Such a "manual restart" procedure can be useful in studies where the simulation is naturally divided into several time stages or phases.

As the first example consider the simulation of circulating cooling water in nuclear reactors where the different stages in the cooling cycle present different reaction conditions like dose rate and temperature. The results from the simulation of the first stage is used as start concentrations in the input file for the second stage, the results from the second stage as input in the third stage, and so on, until the cycle is completed.

Another example is a system that shows sudden changes during the simulation. This may be caused by the depletion of a reactant which is not reformed by any reaction, or by the introduction of a new reactant. Such a rapid change of conditions may slow down or even halt the integration, and restarting from new conditions could be a remedy.

Yet another example is titration as mentioned in Section 11.7.

11.7 Zero-order reactions

We have already seen a special example of zero-order reactions in Section 2.3, where the primary yields were described. In CHEMSIMUL general zero-order reactions can be handled as a special case of exchange equations, cf. Sections 2.5 and 8. In this way we may model a reaction as

$$\frac{\mathrm{d[A]}}{\mathrm{d}t} = k_0 \qquad (78)$$

where k_0 is a constant. When setting up the differential equation for the species A, CHEMSIMUL will then add k_0 to the right-hand side.

Zero-order reactions can be used for simulation of constant-rate addition of a reactant to the reaction system (titration). The setting of start and end time of the titration can be facilitated by means of the `SAVE` command in CHEMSIMUL.

Catalytic reactions can also be described by zero-order reactions. For example, the catalytic decomposition of H_2O_2 can be described by 3 zero-order reactions.

11.8 Third body reactions

Reactions of higher order than 2 should not be considered in condensed phase, and the present version of CHEMSIMUL does not accept reactions of higher order. However, in gas phase many reaction rates are dependent on a "third body" M to absorb surplus reaction energy, as in the reactions

$$\mathrm{A} + \mathrm{B} + \mathrm{M} \rightarrow \mathrm{P} + \mathrm{M} \tag{79}$$

and

$$\mathrm{A} + \mathrm{M} \rightarrow \mathrm{C} + \mathrm{D} + \mathrm{M} \tag{80}$$

with rate constants k_1 and k_2, respectively. These reactions are formally third and second order reactions, but actually they are pseudo-second and pseudo-first order, respectively, because the concentration [M] does not change during the reaction. This means that the differential equation for the third-order reaction

$$\frac{\mathrm{d[P]}}{\mathrm{d}t} = k_1[\mathrm{A}][\mathrm{B}][\mathrm{M}] \tag{81}$$

can be reduced to

$$\frac{\mathrm{d[P]}}{\mathrm{d}t} = k_\mathrm{M}[\mathrm{A}][\mathrm{B}] \tag{82}$$

where

$$k_\mathrm{M} = k_1[\mathrm{M}] \tag{83}$$

To be acceptable for CHEMSIMUL the reactions (79) and (80) must be presented in the following form:

$$\mathrm{A} + \mathrm{B} \rightarrow \mathrm{P} \tag{84}$$

and

$$\mathrm{A} \rightarrow \mathrm{C} + \mathrm{D} \tag{85}$$

These reactions should be assigned the rate constants $k_1 \times [\mathrm{M}]$ and $k_2 \times [\mathrm{M}]$, respectively.

Another option would be to express (81–83) in terms of exchange equations (Chapter 8).

12 Simulation examples

Rasmussen and Bjergbakke [1] gave several examples of simulations using an earlier version of CHEMSIMUL. Their examples spanned from dosimetry and NO_x processes in the atmosphere to the study of radiolysis of ground water from spent nuclear fuel. They also simulated classical processes, notably the Belousov-Zhabotinskii oscillating reaction, which has gained renewed interest as a chemical example of a dynamic system with the typical nonlinear features of limit cycles, period doubling, and eventually chaos.

Here we shall only illustrate two of these test runs with graphical output obtained with CHEMSIMUL plot expressions. Details for these examples are found in Rasmussen and Bjergbakke [1].

12.1 Fricke dosimeter

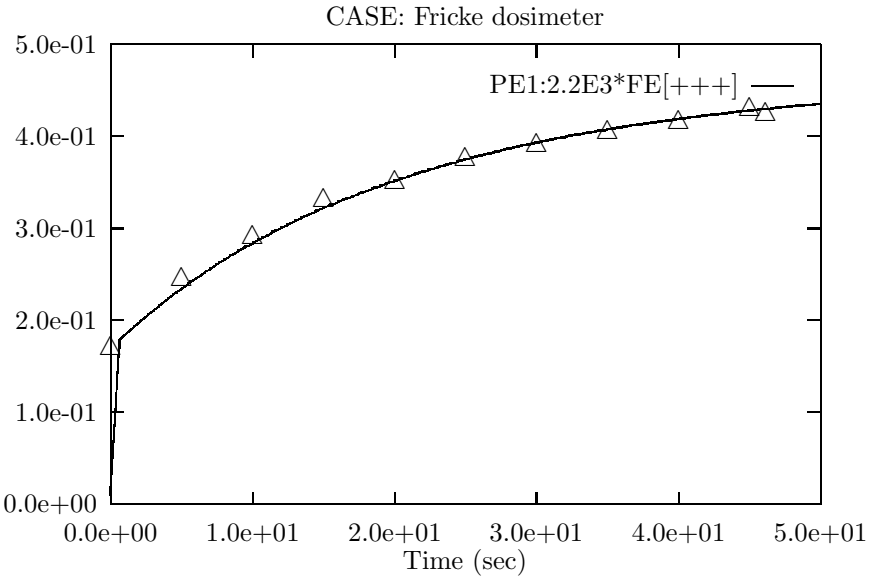

Figure 3. Simulation case: Fricke dosimeter

12.2 Oregonator

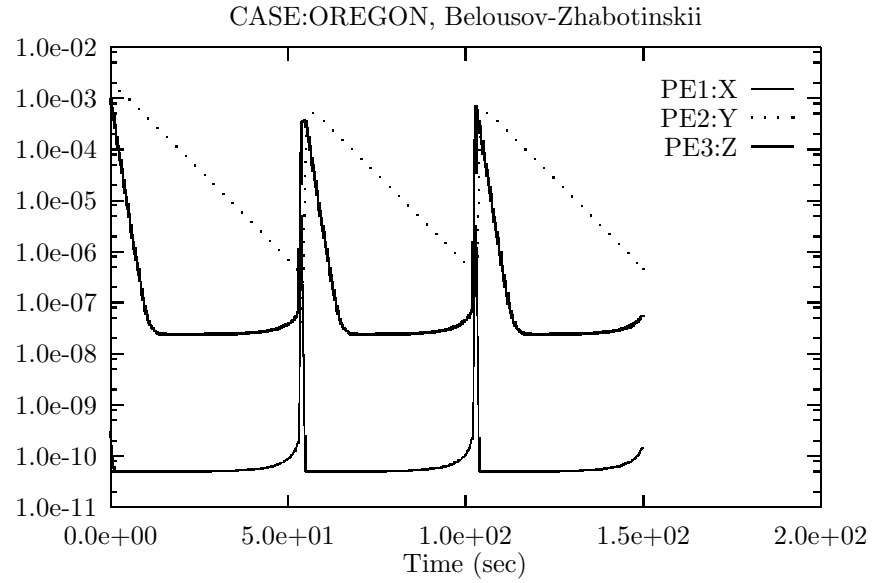

Figure 4. Simulation case: Oregonator

12.3 References to work using CHEMSIMUL

CHEMSIMUL has been used in numerous research projects. To give a few references to relevant publications, we mention the papers by Bouniol and Bjergbakke [14], Christensen and Bjergbakke [15, 16, 17], by Hart *et al.* [18], and by Bjergbakke *et al.* [19, 20].

13 Mathematical details

In the following we shall discuss the principal features in the mathematical implementation of the CHEMSIMUL program. Readers without special interest for the CHEMSIMUL mathematics may skip this chapter.

13.1 Translation principle for reactions

To explain the mechanism of the CHEMSIMUL translator, let us assume a general chemical system being specified with M reactions involving altogether N species. We write the reaction system in the following way:

$$\mathrm{Reac}(r): \quad \alpha_j \times \mathrm{R}_j + \alpha_k \times \mathrm{R}_k \to \sum \alpha_\ell \times \mathrm{R}_\ell, \quad r = 1, \ldots, M \tag{86}$$

where R_s is the chemical symbol for species s, $s = 1, \ldots, N$. The left-hand side is empty for a zero-order reaction. Otherwise α_j, α_k, α_ℓ are non-negative integers satisfying $1 \leq \alpha_j + \alpha_k \leq 2$. For a second-order reaction we may have $j \neq k$ or $j = k$; in both cases $\alpha_j + \alpha_k = 2$. Each reaction $\mathrm{Reac}(r)$ is assigned a rate constant k_r. Moving the reactant terms in (86) to the right-hand side, we may assign with every species s in reaction r a *stoichiometric coefficient* a_{rs}, which is positive for a product and negative for a reactant. Setting unassigned entries $a_{rs} = 0$, we obtain the *stoichiometric matrix* $\mathbf{A} = (a_{rs}) \in \mathbb{Z}^{M \times N}$. This enables us to express the reaction system in compact vector-matrix notation as

$$\mathbf{A}\boldsymbol{v} = \mathbf{0} \tag{87}$$

where the entries of the N-dimensional *species vector* $\boldsymbol{v} \in \mathsf{S}^N$ are the species names R_s in the *symbol set* S for the reaction system. The resulting general ODE system for the species concentrations has the form

$$\frac{\mathrm{d}[\mathrm{R}_s]}{\mathrm{d}t} = \sum_{j,k} n_{jks} M_{jks} [\mathrm{R}_j][\mathrm{R}_k] + \sum_j n_{js} M_{js} [\mathrm{R}_j] + c\, G_s D'(t), \quad s = 1, \ldots, N \tag{88}$$

The right-hand side of (88) is the sum of a quadratic form coming from the second-order reactions, a linear form from the first-order reactions, and a term coming from the zero-order reactions. The 3-dimensional tensors n_{jks} and M_{jks} contain the stoichiometric coefficients a_{rs} and the reaction rates k_r, respectively. For a given s, the sum is taken over pairs (j, k) for which $(\mathrm{R}_j, \mathrm{R}_k)$ is a reacting pair in some reaction $r = r(j, k)$ involving R_s on either side, with $n_{jks} = a_{rs}$ and $M_{jks} = k_r$. A similar interpretation holds for the linear form. For simplicity we here assume that the zero-order term has the form $c\, G_s D'(t)$, where c is the conversion factor, G_s the G-value for species s, and $D'(t)$ the dose rate at time t. (In case of irradiation from decaying isotopes the zero-order term is given by the right-hand side of (43).)

Given a computerized write-up like (25) of the reactions (86), CHEMSIMUL assembles the ODE system (26) or (88) from all the individual pieces in the reactions. On encounter it tabulates every new species and includes its name in the symbol set S. It also stores the rate constants k_r. After the scan phase an assembly phase is invoked. Here CHEMSIMUL constructs a matrix with 5 rows and one column for each second-order or first-order term on the right-hand side of the total ODE system. Entry 2 in the column contains the stoichiometric coefficients n_{jks}, while the other entries contain address pointers to 1) the species s, 3) the reaction r, 4) $[\mathrm{R}_j]$, and 5) $[\mathrm{R}_k]$ (for first-order processes we may set $[\mathrm{R}_k] = 1$). This pointer table gives a complete description of the ODE system and permits the evaluation of its Jacobian.

13.2 Balance check by linear programming

The accurate solution of the ODE system in chemical kinetics requires an overall conservation of the mass balance, as pointed out by Ridler *et al.* [21]. This balance may be affected by two main factors, the reaction mechanism and the integration process. Being a linear multistep method, the numerical integration scheme in CHEMSIMUL has the implicit property of mass preservation, as shown by Rosenbaum [22, 23]. Hence it is sufficient to make a static consistency check, and as stated in Section 5.1, such a check should be based on the demand of existence of strictly positive solutions to the balance equation (28) or (87) where $\mathbf{A}$ is the stoichiometric matrix and $\boldsymbol{v}$ the species vector.

Ridler *et al.* [21] proposed a heuristic checking method based on Gaussian elimination in integer arithmetic on $\mathbf{A}$, and that method was adopted by Rasmussen and Bjergbakke in [1]. Later we replaced it by a mathematically more stringent method: By the homogeneity of (87) we can replace the demand of positive values by a demand of all values be ≥ 1. Thus, if $\mathbf{A} = (a_{ij})$, it must be possible to find solutions $y_j \geq 1$ to the equations

$$\sum_{j=1}^{N} a_{ij} y_j = 0, \quad i = 1, \ldots, M \tag{89}$$

On writing $y_j = x_j + 1$ we may replace (89) by the system

$$\mathbf{A}\boldsymbol{x} = \boldsymbol{b}, \quad \boldsymbol{x} \geq \mathbf{0} \tag{90}$$

where $\boldsymbol{x} = (x_j)$ and $\boldsymbol{b} = (b_i)$ with $b_i = -\sum_{j=1}^{N} a_{ij}$. To decide the feasibility of (90) is a standard linear-programming problem. It can be solved efficiently by the simplex method.

13.3 Solution of the ODE system

A system of ordinary differential equations is conveniently expressed in vector notation:

$$\frac{\mathrm{d}\boldsymbol{y}}{\mathrm{d}t} = \boldsymbol{f}(\boldsymbol{y}, t) \tag{91}$$

The ODE systems in chemical kinetics are typically non-linear and autonomous,

$$\boldsymbol{f}(\boldsymbol{y}, t) = \boldsymbol{f}(\boldsymbol{y}) \tag{92}$$

As mentioned in Section 1.3 the corresponding initial-value problems may be of the stiff type. After many years of experience we are still in favor of using the DLSODA solver from Hindmarsh' ODEPACK collection, written by Petzold [4] and Hindmarsh [5]. This code (originally named LSODA) switches automatically between stiff and nonstiff methods, starting with a nonstiff (Adams) method.

When the stiff BDF procedure is invoked (BDF = Backward Difference Formula), the Jacobian $\mathrm{d}\boldsymbol{f}/\mathrm{d}\boldsymbol{y}$ will be needed. This can be supplied by CHEMSIMUL or estimated by DLSODA, cf. Section 13.4.

DLSODA admits a combined relative and absolute error control: It controls the vector $\boldsymbol{e} = (e_i)$ of estimated local errors in the solution $\boldsymbol{y} = (y_i)$ by the restriction $\|\boldsymbol{r}\|_\infty \leq 1$, where $\boldsymbol{r} = (r_i)$ is given by $r_i = e_i/w_i$ with $w_i = \epsilon_i^{\mathrm{rel}}|y_i| + \epsilon_i^{\mathrm{abs}}$. Thus each component e_i of the local error must satisfy

$$e_i \leq \epsilon^{\mathrm{rel}}|y_i| + \epsilon_i^{\mathrm{abs}} \tag{93}$$

Normally CHEMSIMUL will not need this flexibility; usually fixed relative and absolute tolerances $\epsilon_i^{\mathrm{rel}} = \epsilon_{\mathrm{rel}}$ and $\epsilon_i^{\mathrm{abs}} = \epsilon_{\mathrm{abs}}$, $\forall i$, give good results in practice, and this corresponds to the present implementation.

In radiolytic simulations, where the irradiation burst induces a constant source term in the finite time slot $[0, t_{\mathrm{r}}]$ and then disappears, we found that the best way of using the solver

was to select the one-step mode in combination with a critical time barrier $t = t_r$ (ITASK = 5 in DLSODA). Integration beyond the discontinuity point t_r proceeds *ab initio* from $t = t_r$. The same technique is used for a pulse train. Thus the simulation is partitioned into a number of smooth subproblems.

The one-step mode is also convenient when producing result tables for printing and plotting. Whenever a new printing time t_p is surpassed by the current integration time t, the results at t_p are computed by an interpolation routine provided by DLSODA, cf. Section 13.6. As for the graphics tables, a plotting time mesh is constructed beforehand, either linear or logarithmic, and each time the integration time t enters a new plotting mesh, a plotting point $(t, \boldsymbol{y})$ will be recorded.

There are a number of useful tools and options in DLSODA. It is possible to specify a minimum step length $h_{\min}$ (not implemented in the present CHEMSIMUL version). It is also possible to prescribe a maximum step length $h_{\max}$, and an initial step length h_0 to be attempted instead of the internally computed default value. In the applications we sometimes meet quite hard kinetics problems, where for instance a species approaches zero concentration very rapidly. Here we may see traces of instability with negative concentrations, and even break-down of the integration procedure. A very successful standard remedy for such troublesome cases is to reduce one or more of the parameters in the triple $(\epsilon_{\text{rel}}, h_{\max}, h_0)$.

13.4 The Jacobian

When DLSODA switches to its stiff integration procedure, it will need the Jacobian $d\boldsymbol{f}/d\boldsymbol{y}$ of the right hand side of the ODE system (91). This is a square matrix whose order m equals the dimension of $\boldsymbol{y}$.

The Jacobian needs not to be exact in order to get DLSODA working properly. A fair approximation will do. However, a bad Jacobian approximation may have a devastating influence on the computing time, or even halt the simulation with an integration failure.

There are several ways in which the Jacobian can be supplied. The actual choice is governed by the command `JT = ...` (Section 7.6), as outlined in Table 5:

`JT` value	Kinetics part	Exchange part
1 (default)	Analytic expressions by CHEMSIMUL	Numerical estimate by CHEMSIMUL
2	Numerical estimate by DLSODA	Numerical estimate by DLSODA
3	Analytic expressions by CHEMSIMUL	Contribution ignored

Table 5: Choices for Jacobian evaluation

The numerical estimates are provided by difference quotient approximations.

If the system is dominated by purely kinetic equations (and possibly temperature), with no or few exchange equations (Section 2.5), experience shows that it is best to supply an analytical Jacobian. Since $\mathrm{d}\boldsymbol{f}/\mathrm{d}\boldsymbol{y}$ is then simple and cheap to calculate exactly, this will normally give a faster integration with less risk of error stops.

Many problems require that the Jacobian contribution from the exchange equations is taken into account, in order to make the integration proceed at a reasonable speed. This is why we have made `JT = 1` the default option in CHEMSIMUL.

In other problems the exchange equations have little influence on the Jacobian, and its contribution may safely be ignored. For such cases `JT = 3` is available.

Only in rare cases `JT = 2` should be used. The reason is that the numerical estimation of derivatives in the kinetics part will slow down the calculations, in particular in systems with many chemical reactions.

Contribution to the Jacobian from refreshable parameters are ignored unless `JT = 2`. This approximation is tolerable since RPs are normally slowly varying with time.

In the following we present the proper formulas for the Jacobian elements in the analytical cases `JT = 1` and `JT = 3`, assuming first that the vector $\boldsymbol{y}$ contains concentrations only. In that case $m = N$, i.e. the number of species, and it is easy to compute $d\boldsymbol{f}/d\boldsymbol{y}$ from (88); see also the ODE system (26) for our sample case. We find for the (s,k)-element:

$$\Big(\frac{\mathrm{d}\boldsymbol{f}}{\mathrm{d}\boldsymbol{y}}\Big)_{sk} = \sum_j (n_{jks}M_{jks} + n_{kjs}M_{kjs})[\mathrm{R}_j] + n_{ks}M_{ks} \tag{94}$$

For the interpretation of the j-sum, see similar comments in Section 13.1.

Incorporation of the adiabatic process described in Section 2.7 induces an augmentation of the Jacobian with one extra row and column, such that now $m = N + 1$. Row and column m refer to the temperature T. Let us first consider column m. This will be nonzero if there is a temperature dependence in some of the reaction rates $k_r = k(T)$. For example, with the modified Arrhenius form (5) we find

$$\frac{\partial k}{\partial T} = AT^\beta \exp(-E_\mathrm{a}/(RT))\Big(\frac{\beta}{T} + \frac{E_\mathrm{a}}{RT^2}\Big) \tag{95}$$

Practical experience shows that (95) has negligible influence on the behavior of the numerical integration. The same can be expected when $k(T)$ is determined by tabular interpolation. In these cases the influence from temperature on the parameters is usually much weaker than that coming from the reaction rates. Consequently we can safely set the entire column m to zero. Let us then consider row m. Here we find from (8–9):

$$\Big(\frac{\mathrm{d}\boldsymbol{f}}{\mathrm{d}\boldsymbol{y}}\Big)_{m\ell} = \frac{1}{S_\mathrm{cap}^2}\Big(\Big(\sum_{r=1}^{M} k_r q_r(\delta_{j\ell}[\mathrm{R}_k] + \delta_{k\ell}[\mathrm{R}_j])\Big)S_\mathrm{cap} - Q'(t)c_v(\mathrm{R}_\ell)\Big), \quad \ell = 1,\ldots,N \tag{96}$$

where S_cap is the denominator of (8), and δ_{ij} is the Kronecker delta. The species indices $j = j(r)$ and $k = k(r)$ depend on the reaction r; for first-order reactions r the corresponding summand in (96) should read $k_r q_r \delta_{j\ell}$.

The dose terms in (88) or (43) bear no influence on the Jacobian.

13.5 Computation of derivatives

Derivatives may be requested either in the result table by the command `DERIVATIVE` or as part of a plot expression by the syntax `S'`. It is well known that it is harder to compute fair approximations to the derivative of a quantity in a numerical simulation, than to compute the quantity itself. In this respect CHEMSIMUL is no exception. In principle, we may compute $\mathrm{d}[\mathrm{S}]/\mathrm{d}t$ in two ways:

- Method I: Direct evaluation from the kinetics equation for the species:

$$\frac{\mathrm{d}[\mathrm{S}]}{\mathrm{d}t} = \mathrm{RHS} \tag{97}$$

  where RHS usually contains positive and negative terms involving concentrations and rate constants of `S` and the other species that affect the build-up or decay of `S`. (An example of kinetics equations was presented in the write-up (26) in Section 3.2.)

- Method II: By numerical estimation from the simulated curve $[\mathrm{S}](t)$ produced by CHEMSIMUL either as a result table or preferably a plot table. This may be done by using symmetric difference-quotient approximations to the derivatives.

Both methods have advantages and drawbacks. Method I is readily implemented, as the kinetics equations are needed anyway in the simulation. It gives a "snapshot" computation of d[S]/dt at current time. In the majority of cases it is more accurate than Method II. However, it occasionally breaks down completely, in particular when `S` is an intermediate low-concentration species. It may then happen that RHS in (97) becomes the difference between very large contributions from dominant species. If CHEMSIMUL calculates the latter to a relative precision of say $\epsilon_{\mathrm{rel}} = 10^{-3}$, all precision may get lost by cancellation so that d[S]/dt will contain numerical noise only.

Although Method II is usually less accurate it is more robust in extreme situations. It may seem a paradox that taking quotient differences in the solution curve could really be better than using the kinetics equations directly; after all the latter equations are the basic input for the ODE solution. The paradox is partly resolved by considering that DLSODA uses advanced techniques with good stability and conservation properties, and stores important information on the behaviour of the solution in a neighbourhood of the current simulation time. This admits a reasonable computation even of the less significant species.

CHEMSIMUL uses Method I for derivatives in result tables, and Method II for derivatives in plot expressions.

13.6 Interpolation techniques

Various kinds of interpolation techniques are used in CHEMSIMUL. For interpolation in the user-specified tables (Sections 7.14.2 and 7.14.3) we use cubic splines. There are several variants of such splines. The *natural spline* is characterized by vanishing second derivatives at the end points T_1 and T_n. We prefer however the *Moler spline* [11], where the third derivative in T_1 matches that of the unique cubic through T_1, T_2, T_3, and T_4 and similarly at the other end.

Prediction of maximum concentrations and correponding time postions (Section 5.3) is facilitated by parabolic interpolation over 3 points picked from the "plot table" which is calculated during integration. This parabola corresponds to the Lagrangian polynomial for the 3 points.

Also the DLSODA integrator [4, 5] uses interpolation. This is based on the so-called Nordsieck history array.

13.7 Implementation of regular expressions

When describing the implementation of regular expressions (cf. Section 6.4) we use a plot expression example:

```
PE1: 4711.77 + (log10(OH[-])/7.45)^1.5
```
(98)

The interpretation of such an expression has two phases, a lexicographical analysis and a successive processing. The lexicographical analysis decomposes the plot expression into a list of *atoms*. Each atom is a constant, a species name, an arithmetic operator, or a function name. The 7 possible operators are those of elementary algebra: $*$, /, $+$, $-$, `^`, (, and), where `^` stands for exponentiation; depending on context $+$ and $-$ can be infix or unary. The 10 standard mathematical functions cos, cosh, exp, ln, $\log_{10}$, sin, sinh, $\sqrt{\ }$, tan, and tanh are supported. The string of atoms is ended with an "empty atom", which is considered as an extra operator. In the above example we get the atom list shown in Table 6, where species are supposed to be assigned definite concentration values:

Atom#	Atom	Type
1	`4711.77`	Constant
2	`+`	Operator
3	`(`	Operator
4	`log10`	Function
5	`(`	Operator
6	`OH[-]`	Species
7	`)`	Operator
8	`/`	Operator
9	`7.45`	Constant
10	`)`	Operator
11	`^`	Operator
12	`1.5`	Constant
13	empty	Operator

Table 6. Atoms in a plot expression

We process this table atom by atom using common techniques from computer science: We create and maintain two stacks **V** and **O**, the first for values (operands) and the second for operators. Moreover we use a separate stack **F** for the mathematical function names. Functions are considered as unary operators of highest priority with the common class name "f".

An operand may be a constant, a concentration value, or a computation result. A new operand always pushes **V** and enters its top.

Let N be a new operator and T the current top-of-stack operator. If $N =$ "(" $\vee N =$ "f", N pushes **O** and enters its top; when $N =$ "f" it also pushes **F**. Otherwise we continue from the algorithmic point called A: If $|\mathbf{O}| = 0$, N pushes **O**. But if $|\mathbf{O}| > 0$ we observe whether $T =$ "(" $\wedge N =$ ")". If so, T is devoured and **O** is popped. If not we continue by comparing priorities π: If $\pi(N) \leq \pi(T)$, a unary or binary operation takes place on the current top value(s). If $T =$ "f", a function call is executed followed by popping the function stack; if T is unary $+$ or $-$ the corresponding operation takes place; otherwise we have a binary operation with $T = +$, $-$, $*$, $/$, or $\hat{}$. After the operation **O** is popped, while the top value is updated with the current result. If the operation was binary, **V** is popped, too. After popping **O** we go back to point A in the algorithm. If, on the other hand, $\pi(N) > \pi(T)$, then **O** is pushed, and so is **F** if $N =$ "f". After storing the operator N, a new atom will be processed.

A small complication arises when repeated exponentiation occurs, since the operation order is then from right to left: When $N = T = \hat{}$, we postpone the binary operation simply by letting $\hat{}$ push **O** and processing immediately to the next atom.

The algorithm terminates when the empty operator enters the top of **O**. When this happens we must have $|\mathbf{V}| = 1$.

We use the following operator priority list (from low to high):

$$\{\text{empty}\}, \quad \{(\ ,\ )\} \quad \{\text{infix}+,\ \text{infix}-\}, \quad \{\times,\ /\}, \quad \hat{}, \quad \{\text{unary}+,\ \text{unary}-\}, \quad \{\text{f}\} \qquad (99)$$

The rules given above are fairly standard for processing expressions in programming languages, mathematical systems, and calculators. Quite general formulas can be treated in this way, like for instance

$$(+A + 0.19 \times B \times (B-3) \times \sin(C) \times Q - \sqrt{J/40}) \times (-E/F \times 0.2 + 1/G\hat{}3) \qquad (100)$$

13.8 Solution of equations from isotope filiation

Linear isotope families
Consider first a *linear isotope family* which is a chain of decaying nucleides p_1, p_2, ..., p_{n-1}, p_n, where p_1 decays by some nuclear process with decay rate k_1 and thereby produces p_2. Similarly p_2 transforms to p_3 with the decay rate k_2, and so on; finally p_{n-1} disintegrates with the decay rate k_{n-1} to p_n which is a stable isotope with $k_n = 0$. Thus we have the following chain

$$p_1 \xrightarrow{k_1} p_2 \xrightarrow{k_2} p_3 \xrightarrow{k_3} \cdots \xrightarrow{k_{n-2}} p_{n-1} \xrightarrow{k_{n-1}} p_n \tag{101}$$

The *length* of the chain is the number of decay processes i.e. $n-1$. We want to compute the amount $y_i(t) = N_i(t)$ of isotope p_i at time t, given the initial amount $y_i(0) = y_{i0}$, for $i = 1, \ldots, n$. Defining $k_0 = 0$ and $y_0(t) = 0$, the amounts $y_i(t) = N_i(t)$ satisfy the linear ODE system

$$\frac{\mathrm{d}y_i}{\mathrm{d}t} = k_{i-1}y_{i-1} - k_i y_i, \quad i = 1, \ldots, n \tag{102}$$

with the initial conditions

$$y_i(0) = y_{i0}, \quad i = 1, \ldots, n \tag{103}$$

Writing $(y_i) = \boldsymbol{y}$ we can express (102) in vector-matrix notation as $\mathrm{d}\boldsymbol{y}/\mathrm{d}t = \mathbf{A}\boldsymbol{y}$, where the constant matrix $\mathbf{A}$ is the Jacobian of the system. It is of order n and lower triangular banded with $a_{ij} = 0$ for $i > j+1$. The eigenvalues are $-k_1, \ldots, -k_n$ (where $k_n = 0$). The vector form of the solution could immediately be stated as $\boldsymbol{y} = \exp(t\mathbf{A})\,\boldsymbol{y}_0$ where $\boldsymbol{y}_0 = (y_{10}, y_{20}, \ldots, y_{n0})^{\mathrm{T}}$. However, in this study we prefer to work directly with the system (102). In the simple case where all the rate constants k_i are distinct, analytical solutions can be computed recursively for $i = 1, \ldots, n$. By induction it can be shown that

$$y_i(t) = \sum_{j=1}^{i} c_{ij} e^{-k_j t}, \quad i = 1, \ldots, n \tag{104}$$

where the coefficients for $i = 1, \ldots, n$ are given by the rules

$$c_{ij} = \frac{k_{i-1}}{k_i - k_j} c_{i-1,j} \ (j < i), \quad c_{ii} = y_{i0} - k_{i-1} \sum_{j=1}^{i-1} \frac{c_{i-1,j}}{k_i - k_j} \tag{105}$$

Use of Laplace transforms for dealing with multiple eigenvalues
We see that (105) breaks down when some of the k_i coincide. Though this case is not very likely to occur in practice, it will be covered, too. An appropriate tool for making this generalisation is the method of *Laplace transforms*. Thus we put

$$\mathrm{L}[y_i(t)] = Y_i(s), \quad i = 1, \ldots, n \tag{106}$$

where the operator L is defined by $f(t) \mapsto F(s)$ with

$$F(s) = \int_0^\infty e^{-st} f(t)\mathrm{d}t \tag{107}$$

The counterpart to (102) becomes the system

$$-k_{i-1}Y_{i-1}(s) + (s + k_i)Y_i(s) = y_{i0}, \quad i = 1, \ldots, n \tag{108}$$

which may be put on the recursive form

$$Y_i(s) = \frac{y_{i0} + k_{i-1}Y_{i-1}(s)}{s + k_i} \tag{109}$$

From this we readily find by induction:

$$Y_i(s) = \frac{1}{s + k_i} \sum_{r=1}^{i} y_{r0} \prod_{j=r}^{i-1} \frac{k_j}{s + k_j}, \quad i = 1, \ldots, n \tag{110}$$

The validity of (108-110) is not impaired by coinciding decay rates. Let $\kappa_1, \kappa_2, \ldots$ be the *distinct* values of $\{k_j\}_{j=1}^n$, in order of occurrence. Then the common denominator in (110) takes the form

$$\prod_{\lambda=1}^{\mu_i} (s+\kappa_\lambda)^{\alpha_{i\lambda}} \tag{111}$$

where $\alpha_{i\lambda}$ is the multiplicity of κ_λ in $\{k_j\}_{j=1}^i$ and $\mu_i = |\{k_1, \ldots, k_i\}|$. The next step is to expand (110) in partial fractions corresponding to (111):

$$Y_i(s) = \sum_{\lambda=1}^{\mu_i} \sum_{r=1}^{\alpha_{i\lambda}} \frac{c_{i\lambda r}}{(s+\kappa_\lambda)^r} \tag{112}$$

Having obtained (112) we return to time domain by the inversion law

$$\mathrm{L}^{-1}\left[\frac{1}{(s-c)^r}\right] = \frac{t^{r-1}e^{ct}}{(r-1)!} \tag{113}$$

The proper tools for calculating the coefficients $c_{i\lambda r}$ in terms of $c_{i-1,\lambda r}$ are (109) and the identity

$$\frac{1}{(s-a)^k(s-b)} = \frac{1}{(b-a)^k}\frac{1}{s-b} - \sum_{j=1}^{k} \frac{1}{(b-a)^{k-j+1}}\frac{1}{(s-a)^j} \qquad (a \neq b) \tag{114}$$

(Proof: Consider first $a = 0$. This case is easily established by multiplication with $s^k(s-b)$; now (114) follows by substituting $s := s-a$, $b := b-a$ □.) We must distinguish between the case where k_i in (109) differs from all the rate constants κ_λ, $\lambda = 1, \ldots, \mu_{i-1}$, and the case where there is a coincidence with one of these, $k_i = \kappa_\lambda$. In order to compute accumulated doses we shall furthermore need the integrals

$$w_i(t) = \int_0^t y_i(\tau)\mathrm{d}\tau \tag{115}$$

Their Laplace transforms are simply

$$W_i(s) = \frac{Y_i(s)}{s} \tag{116}$$

When (112) is inserted in (116) we can again use (114), now with $b = 0$, to obtain an expression of the same kind as (112). (Note that $\kappa_\lambda > 0$ unless the isotope is stable.)

The generalized filiation problem

Until now we have assumed that each isotope family is a simple linear chain (101). However we shall need a more general model that can deal with isotopic branching and other irregularities. The extended model treats the decaying isotopes as a general system of first-order reactions $p \to q$. In order to derive the Laplace transforms of the current amounts of the isotopes we found it useful to represent the nuclear reactions by a directed graph Γ with vertex set $V(\Gamma)$ and arc set $A(\Gamma)$. Each vertex $p \in V(\Gamma)$ identifies an isotope and each arc $j = pq \in A(\Gamma)$ corresponds to the disintegration of isotope p into isotope q with decay rate k_j. In this model each component of Γ represents an isotope family. Cycles in Γ are excluded, which means that no isotope can, directly or indirectly, disintegrate into itself. Vertices with zero out-degree represent stable isotopes. The previously described calculation scheme can now be generalized in a straightforward manner. Let us number the isotopes including the stable ones arbitrarily from 1 to m, thus identifying each vertex $p \in V(\Gamma)$ with an integer from $\{1, \ldots, m\}$. First we should replace the ODE (102) by

$$\frac{\mathrm{d}y_p}{\mathrm{d}t} = \sum_{j \in J_{\mathrm{in}}(p)} k_j y_{q_j} - \ell_p y_p, \quad p = 1, \ldots, m \tag{117}$$

where

$$\ell_p = \sum_{j \in J_{\text{out}}(p)} k_j \tag{118}$$

Here $J_{\text{in}}(p)$ is the set of incoming arcs to vertex p, while $J_{\text{out}}(p)$ is the set of outgoing arcs; the subscript q_j denotes the starting vertex of arc j. The initial conditions are as in (103). The modified version of (109) reads

$$Y_p(s) = \frac{y_{p0} + \sum_{j \in J_{\text{in}}(p)} k_j Y_{q_j}(s)}{s + \ell_p} \tag{119}$$

The necessary modifications of the previous analysis of the linear chain are straightforward. In particular, when all the ℓ_p are distinct, (119) takes the form

$$Y_p(s) = \sum_{q=1}^{m} \frac{c_{pq}}{s + \ell_q}, \quad p = 1, \ldots, m \tag{120}$$

If some of the ℓ_p coincide we get an expression like (112). The correct flow of the generalized Laplace calculations is ensured by a proper traversal order of Γ. This will be described in Section 13.10, which also mentions the need of *processing* an arc j from $q = q_j$ to p. In the present case this means recording the Laplace term $k_j Y_{q_j}(s)$ in (119) for the vertex p. Moreover, the criterion for shifting the mark $\alpha(p)$ to OLD given in Section 13.10 ensures that $Y_p(s)$ is fully evaluated before its effects on the descendants of p are assessed. The action mentioned in step c) of the algorithm is here the completion of the Laplace transform $Y_p(s)$ by (119).

The computations in (119) are facilitated by the identity (114).

13.9 Refreshable parameters

In Section 7.11 we described how the user could work with *refreshable parameters* (RPs) in CHEMSIMUL. We shall here give some details of our computational scheme for evaluating a given set of RPs.

As mentioned in Section 7.11 it is essential that each RP is given in *explicit form* as a functional expression in other RPs together with symbolic constants and current concentration values etc. Ignoring here the latter contributions, we are thus given an RP system of the form

$$z_i = f_i(z_k, \ldots, z_\ell), \qquad i = 1, \ldots, m \tag{121}$$

(In contrast to this, an *implicit* equation in the RPs is an equation of the form

$$g_i(z_k, \ldots, z_\ell) = 0 \tag{122}$$

Such equations are more difficult to handle and are excluded from CHEMSIMUL.)

Plugging in the newly evaluated values of the RPs into the ODE system will normally cause no great perturbation of the latter. However, in the computation of the Jacobian (cf. Section13.4) the RP contribution is ignored unless `JT = 2`. This may slow down the convergence rate, but we expect that this effect is of minor importance since RPs are typically slowly varying with time.

The most difficult part in our RP computational algorithm was the design of an efficient and reliable method to order the RPs in such a way that each new RP depends only on already evaluated RPs. In fact our goal was to find a permutation

$$\pi: \quad i \mapsto \pi_i, \qquad i = 1, \ldots, m \tag{123}$$

that accomplishes this task. To do this we represent the RP equation set (121) by a directed graph Γ with vertex set $V(\Gamma)$ and arc set $A(\Gamma)$. Each vertex $p \in V(\Gamma)$ identifies an RP z_i

of LHS(121), while each RP on RHS(121), say z_k, generates an arc $j = pq \in A(\Gamma)$ going from z_k to z_i. Each component of Γ represents a separate resolvable chain of RPs which can be evaluated in turn. Cycles in Γ are excluded, which means that no RP can reference itself, directly or indirectly. We assign a unique vertex $p = p_i \in V(\Gamma)$ to each RP $= z_i$ of LHS(121).

A correct flow of the RP calculations is ensured by using the graph traversal algorithm described in Section 13.10, which also mentions the need of *processing* an arc j from $q = q_j$ to p. This is here the recording of the specific argument z_k in (121) that corresponds to q and enters the computation of the RP z_i corresponding to p. The action mentioned in step c) of the algorithm is here the identification of the next component of the permutation π in (123). When the algorithm terminates, the permutation π is completely determined.

13.10 Graph-theoretical methods

It is interesting that the general isotope filiation problem in Section 13.8 and the ordering principle for refreshable parameters in Section 13.9 are solved successfully by the same graph-theoretical algorithm, which we shall now describe. In both cases the calculations are organised by traversing a *digraph* (directed graph) Γ with vertex set $V(\Gamma)$ and arc set $A(\Gamma)$. The interpretation of vertices and arcs depends on context.

We must devise a proper traversal order of Γ, in which we shall exclude cycles. Our graph representation is based on *double neighbor lists*, where each vertex p has a list of incoming arcs qp and a list of outgoing arcs pq. The traversal algorithm uses two *state functions* $\alpha : V(\Gamma) \to \{\text{NEW}, \text{OLD}\}$ and $\beta : A(\Gamma) \to \{\text{NEW}, \text{OLD}\}$ defined on the set of vertices and arcs, respectively. Both are dynamic in the sense that they change as the algorithm proceeds. By *processing* an arc j from $q = q_j$ to p we mean making some context-dependent operation as detailed in Section 13.8 or 13.9. Initially $\alpha(p) = \beta(j) = \text{NEW}$ for all $p \in V(\Gamma)$ and all $j \in A(\Gamma)$. When an arc j is processed we set $\beta(j) = \text{OLD}$. *It is a crucial principle that arcs $j = pq$ should not be processed before all incoming arcs towards p are marked as* OLD. When this is detected the mark $\alpha(p)$ shifts to OLD. Vertices with zero in-degree represent valid starting points for identifying a component of Γ. We can now outline the following steps of the algorithm:
a) Initialization: Mark all vertices and arcs as NEW. Select an initial vertex p. Proceed to b).
b) See if there is a NEW arc $j = qp$ towards p. If yes, backtrack to q, set $p := q$ and reiterate from b). If no, proceed to c).
c) If p is NEW, then we perform a context-dependent calculation step. We mark p as OLD and proceed to d). Otherwise go directly to d).
d) See if there is a NEW arc $j = pq$ from p. If yes, we process the arc, mark it as OLD, let $p := q$, and return to b). If no, proceed to e).
e) If there is a previous vertex q, backtrack to this, let $p := q$, and go back to d). If not proceed to f).
f) Another graph component has been completed. If there are no more NEW vertices, the algorithm is finished. Otherwise select a NEW vertex p and return to b).

The algorithm terminates when there are no more NEW vertices.

14 Updates since previous versions

CHEMSIMUL has undergone many revisions and enlargements since the first official 1984 edition documented by Rasmussen and Bjergbakke [1].

The first CHEMSIMUL version for PC dates back to 1992-1993. By and large, this was a FORTRAN translation of the 1984 ALGOL code, with few new facilities, notably the modified Arrhenius rate constants (Section 2.6) and inclusion of adiabatic processes (Section 2.7).

14.1 Updates 1994-1998

- The program is made independent of specific units.
- Mass balance check made mathematically correct by linear programming.
- The diagnostics for input errors are improved and made more precise.
- There is no limit on the number of reactions.
- There is no limit on the number of species.
- There is no limit on the number of terms in differential equations.
- Decaying isotopic radiation can be handled.
- There is a test for electro neutrality for each reaction.
- There is a test for electro neutrality for the totality of G-values.
- Gas constant R entered in input file when needed.
- Conversion factor c may be modified in the input.
- Error in the analytical Jacobian for adiabatic processes was corrected.
- Case-sensitive interpretation of species names.
- Year 2000 protection.

14.2 Updates 1999

- A train of pulses can be handled by the new command `NRR`.
- Heat production from external and isotope irradiation included.
- The program may compute and print integrated doses from decaying isotopes.
- Isotope identification need no longer be numeric, but can also be alphabetic.
- Decaying mixed-particle input is included in the input echo.
- Comment lines made possible also within Isotope Block.
- Exponentiation operator is allowed in plot expressions.
- Mathematical functions in plot expressions: cos, exp, log, $\log_{10}$, sin, $\sqrt{\ }$.
- Improved layout of input echo in result file; redundant items removed.
- Progress bar on screen showing progress of simulation.
- Command-line input possible for PC version, e.g. `chem h2-o2`.

- New command `UNBAL` for overruling rejection of unbalanced systems.
- `MODE` command for choosing between adiabatic and isothermal simulation.
- Problem case text may be continued on a second line.
- Mathematical integration command `EPS` is no longer needed.
- Mathematical integration command `FSTSTP` is no longer needed.
- Mathematical integration command `HMAX` is no longer needed.
- `PRINTS` no longer required (now only `TEND` required).
- Negative results no longer suppressed in result file.

14.3 Updates 2000

- Preventing excessive output volume when using the command `NRR`.
- Possibility of restricting printout to selected species.
- Derivative tables can be produced in result file.
- Plot expressions may contain derivatives.
- Time variable `TIME` can enter as a primary in plot expressions.
- Restart file for concentrations produced automatically.
- Adjustable width of main table and symbolic ODE output.
- New output table: dose rates from isotopes.
- New output table: Specific dose rates by radiation and isotopes.
- Dose rate from isotopes can enter as a primary in plot expressions.
- Restart procedure for isotopes dose rates implemented.
- After an integration failure the simulation time until the halt will be printed.
- Conversion factor c echoed to result file when relevant.
- Gas constant R echoed to result file when relevant.

14.4 Updates 2001

- Prompts for experimental plot file only when command `EXPDATA` is present.
- Primary yields may depend on absolute temperature by polynomial functions.
- New design for radiolysis from decaying isotopes: general isotope families.
- Optional heat production specifier `heat=` introduced for nuclear reactions.
- General zero-order reactions implemented.
- The variation of reaction rates with temperature can be specified by tables.
- G-values no longer checked for electro neutrality.
- `MAXCON` gives only max concentrations; formation/decay half-lives dropped.
- The interpolation procedure in the `MAXCON` evaluation is improved.
- Command `MODE` for choosing adiabatic/isothermal simulation deimplemented.

- Command `NORMALIZE` for plot normalisation is deimplemented.
- Multiple reactions with same identification number no longer allowed.
- Repeated input of `CON(...)` with same reactant no longer allowed.
- Repeated input of `G(...)` with same reactant no longer allowed.
- Repeated input of `HCV(...)` with same reactant no longer allowed.
- Restart procedure replaced by simplified dump/restart mechanism.
- New command `NOTABLE` for suppressing output of result tables.
- Enhanced input check with rejection of negative values and duplicates.
- Redesign of output: Now clearly divided in sections including input echo.

14.5 Updates 2002

- Isotopic part of CHEMSIMUL can be used without chemical simulation.
- 3-term syntax for isotopic `AMOUNT` deleted.
- `ACTIVITY` replaces `AMOUNT` in the isotopic input.
- A new isotope command `DECAYPRINT` tabulates all amounts and activities.
- Heat production specifier `heat=` for nuclear reactions deimplemented.
- Labelled chemical reactions with empty contents no longer allowed.
- Plot command filename changed to `chemgnu.plt` to match *gnuplot*.
- Identifiers for nuclear reactions must now include a number as `NR1:`.
- In case of error program stays in command window for inspecting message.
- Interface to ODE solver upgraded to 1997 version of LSODA.
- The names `TEMP` and `TIME` will be rejected as species names.

14.6 Updates 2003

- Exchange equations introduced for describing transport phenomena.
- Improved Jacobian evaluation for exchange equations combined with kinetics.
- Empty input file now implies "do nothing" rather than an error stop.
- $x < 0$ in x^y or $\sqrt{x}$ in plot/exchange equations $\Rightarrow$ warning *after* simulation.
- `d[...]/dt` changed to `d(...)/dt` in `DIFFEQ` prints.
- Zero-order command deleted; obsoleted by exchange equations.
- Third-body command `THIRD` deleted; obsoleted by exchange equations.
- Salt effect included: impact of ionic strength on rate constants.
- Renaming of the natural logarithm function from `log` to `ln`.
- Symbolic constants introduced, may be used in exchange equations.
- License key and license file introduced.

14.7 Updates 2004

- Interface to ODE solver upgraded to 2003 version, now called DLSODA.
- Time variable `TIME` can enter as a primary in exchange equations.
- Command `CONVERT` can enter as a primary in plot/exchange equations.
- Command `TEND` can enter as a primary in plot/exchange equations.
- Command `IONIC` can enter as a primary in plot/exchange equations.
- `DUMP`/`RSTART` mechanism replaced by `SAVE` files and manual restart.
- Isotope Block in `SAVE` file with final dose coefficients `DA=`... etc.
- Isotope Block in `SAVE` file with final activities.
- Radiolytic G-values can occur as primaries in plot/exchange equations.
- Nuclear G-values can occur as primaries in plot/exchange equations.
- Total isotopic doserates by emission type in plot/exchange equations.

14.8 Updates 2005

- Reorganised and simplified integration procedure for pulses.
- `NOTABLE` command deimplemented; use instead `PRINTS = 0` (or omit `PRINTS`).
- Syntax `TEMP TABLE RE`... changed to `TABLE TEMP RE`...
- Imposed temperature variation $T(t)$ with time by interpolation table.
- Fixed or imposed temperature can enter plot expressions.
- Fixed or imposed temperature can enter exchange expressions.
- Primary yields as polynomial functions of temperature also for isotopic case.
- Table results below 10^{-99} are truncated to zero.

14.9 Updates 2006

- European comma rejected as decimal point in input numbers.
- Default relative integration precision changed from 10^{-3} to 10^{-5}.
- Default absolute temperature changed from 300 to 298.15.
- New Jacobian option: Numerical estimation by solver.
- New Jacobian option: Contribution from exchange equations ignored.
- Trailing comments after commands are available using `!`
- CHEMSIMUL input reorganized in 13 blocks with `$` headers.
- CASE identification command replaced by Identification Block.
- Trailing record `ENDDATA` should now be `$ ENDDATA`.
- Tables are placed in a Tabular Data Block rather than after `$ ENDDATA`.
- Refreshable Parameters (RP) Block introduced, and RPs made operational.

- Improved and reorganized input echo, including RPs.
- Improved and reorganized problem statistics.
- Exchange equations are allowed to be additive.
- New mathematical functions in expressions: cosh, sinh, tan, tanh.
- Imposed dual temperature variation $(T(t), T_2(t))$ by interpolation table.
- Regular expressions unify plot expressions, exchange equations, and RPs.
- Reaction rates can be set equal to RPs.
- Primary yields can be set equal to RPs for radiolytic case.
- Primary yields can be set equal to RPs for isotopic case.
- Uniform naming convention for species, RPs, and symbolic constants.
- Polynomial primary yields in T deimplemented; obsoleted by RPs.
- Old salt effect procedure deimplemented.
- Molar masses recognized by CHEMSIMUL.
- Automatic computation of the RPs `ION`, `SIGMACi`, and `SIGMAMiCi`.
- `TEND` is no longer required; by default it is zero.
- Miscellaneous minor improvements and error corrections.

14.10 Updates 2007

- Initial time `TSTART` introduced.
- `CONVERT` may be set equal to refreshable parameter.
- For pulse trains `TENDRAD` is replaced by `TTRAIN`.
- `TSTART` can be used in expressions as braced item.
- Licensing is now identical to that in the GUI version.
- Default $c =$ `CONVERT` changed to $1.036427 \cdot 10^{-7}$.
- Heat calculation from pulse irradiation deleted.
- Default gas constant `R` given the value 8.314472.
- Dose unit selector `DOSEUNIT` introduced for *Gy* or *krad*.
- Restricted demo version now only available with GUI.

14.11 Updates 2008

- Substantial improvements in GUI version and many new features.
- Repeated exponentiations like *a^b^c* allowed in regular expressions.
- Parantheses in species names introduced.
- Parantheses in species names can be nested.
- Graphical output possible even without chemistry.
- Always report the state of balance and allowance to continue.

- Plot expression numbers need no longer be consecutive.
- `PLOT` commands introduced, making `PLONPA` obsolete.
- `XLOG` and `YLOG` options in `PLOT`, making `OUTTYPE` obsolete.
- Logarithmic time scaling deimplemented in Graphics Table File.
- Optional grid in plots.
- Optional curve marks in plots.
- Possibility of cross plots of plot expressions without time.
- Extra headline with PEs in Graphics Table File.
- General data file plotting replaces `EXPDAT` plotting of experimental data.
- Classical version now runs without licensing.
- New command `PRINTONLY_RP` for restricted printing of RPs.
- New storage design permits very long regular expressions.
- Miscellaneous minor improvements and error corrections.

References

[1] O. L. Rasmussen and E. Bjergbakke, "CHEMSIMUL — a program package for numerical simulation of chemical reaction systems," Tech. Rep. Risø-R-395(EN), Risø National Laboratory, DK-4000 Roskilde, Denmark, January 1984. Available in pdf from CHEMSIMUL home page www.chemsimul.dk.

[2] C. W. Gear, "Algorithm 407, DIFSUB for solution of ordinary differential equations," *Commun. ACM*, vol. 14, p. 185, 1971.

[3] A. C. Hindmarsh and G. D. Byrne, "EPISODE: An experimental package for the integration of systems of ordinary differential equations," Tech. Rep. UCID-30112, Livermore, 1975.

[4] L. R. Petzold, "Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations," *SIAM J. Sci. Stat. Comput.*, vol. 4, pp. 136–148, 1983.

[5] A. C. Hindmarsh, "ODEPACK, a systematized collection of ode solvers in scientific computing," in *Scientific Computing* (R. S. Stepleman, ed.), (Amsterdam), pp. 55–64, North-Holland, 1983.

[6] M. Metcalf and J. Reid, *FORTRAN 90/95 explained.* Oxford: Oxford University Press, 1996.

[7] D. R. Lide, *(ed.), CRC handbook of chemistry and physics, 84th ed., 2003-2004.* Boca Raton: CRC Press, 2003.

[8] O. L. Rasmussen, E. Bjergbakke, P. Pagsberg, and P. Kirkegaard, "CHEMSIMUL — software and methods for numerical simulation of chemical reaction systems," in *Industrial Mathematics Week*, (Trondheim), pp. 174–181, Norwegian Institute of Technology (NTH), 1992. Paper presented by P. Kirkegaard.

[9] T. Williams and C. Kelley, *GNUPLOT — An Interactive Plotting Program*, 2002. Online Manual for Version 3.7.

[10] U. M. Ascher and L. R. Petzold, *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. Philadelphia: SIAM, 1998.

[11] G. E. Forsythe, M. A. Malcolm, and C. B. Moler, *Computer methods for mathematical computations*. Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1977.

[12] C. W. Davies, "Salt effects in solution kinetics," in *Progress in reaction kinetics* (G. Porter and B. Stevens, eds.), vol. 1, pp. 161–186, Pergamon Press, 1961.

[13] C. W. Davies, "The extent of dissociation of salts in water. Part VIII. An equation for the mean ionic activity coefficient of an electrolyte in water, and a revision of the dissociation constants of some sulphates," *J. Chem. Soc.*, vol. 2, pp. 2093–2098, 1938.

[14] P. Bouniol and E. Bjergbakke, "A comprehensive model to describe radiolytic processes in cement medium," *Journal of Nuclear Materials*, vol. 372, pp. 1–15, 2008.

[15] H. Christensen and E. Bjergbakke, "Alfa-radiolysis of aqueous solutions," in *Mat. Res. Soc. Symp. Proc.*, vol. 50, p. 401, Materials Research Society, 1986.

[16] H. Christensen and E. Bjergbakke, "Application of CHEMSIMUL for groundwater radiolysis," *Nuclear and Chemical Waste Management*, vol. 6, p. 265, 1986.

[17] H. Christensen and E. Bjergbakke, "Radiation induced dissolution of UO2," in *Mat. Res. Soc. Symp. Proc.*, vol. 84, (Boston), Materials Research Society, 1987.

[18] E. J. Hart, K. Sehested, E. Bjergbakke, and J. Holcman, "Gamma-ray initiated chain decomposition of aqeuous ozone solutions," *Radiat. Phys. Chem.*, vol. 29-5, p. 399, 1987.

[19] E. Bjergbakke, Z. D. Draganic, K. Sehested, and I. G. Draganic, "Radiolytic products in waters. Part I: Computer simulation of some radiolytic processes in the laboratory," *Radiochimica Acta*, vol. 48, p. 65, 1989.

[20] E. Bjergbakke, Z. D. Draganic, K. Sehested, and I. G. Draganic, "Radiolytic products in waters. Part II: Computer simulation of some radiolytic processes in nature," *Radiochimica Acta*, vol. 48, p. 73, 1989.

[21] G. M. Ridler, P. F. Ridler, and J. G. Sheppard, "A systematic method of checking of systems of chemical equations for mass balance," *J. Phys. Chem.*, vol. 81, p. 2435, 1977.

[22] J. S. Rosenbaum, "Conservation properties of numerical integration methods for systems of ordinary differential equations," *J. Comp. Phys.*, vol. 20, p. 259, 1976.

[23] J. S. Rosenbaum, "Conservation properties of numerical integration methods for systems of ordinary differential equations. 2.," *J. Phys. Chem.*, vol. 81, p. 2362, 1977.

Index